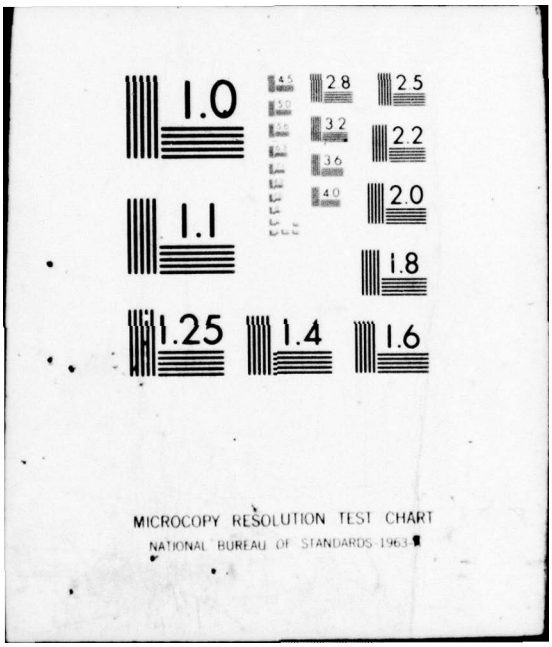
AD-A036 116 SYSTEM DEVELOPMENT CORP SANTA MONICA CALIF F/G 9/2
SOFTWARE DATA COLLECTION STUDY. VOLUME II. AN ANALYSIS OF SOFTW--ETC(U)
DEC 76 N E WILLMORTH, M C FINFER F30602-75-C-0248
UNCLASSIFIED SDC-TM-5542/002/01 RADC-TR-76-329-VOL-2 NL

1 OF 2
AD
A036116


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

13

RADC-TR-76-329, Volume II (of eight)
Final Technical Report
December 1976

SOFTWARE DATA COLLECTION STUDY
An Analysis of Software Data Collection Problems and Current Capabilities

System Development Corporation

ROME AIR DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
GRIFFISS AIR FORCE BASE, NEW YORK 13441

This report contains a large percentage of machine-produced copy which is not of the highest printing quality but because of economical consideration, it was determined in the best interest of the government that they be used in this publication.

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public including foreign nations.

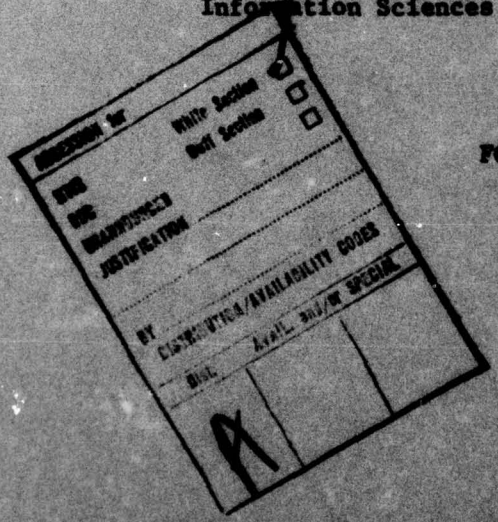
This report has been reviewed and is approved for publication.

APPROVED: Richard T. Slavinski

RICHARD T. SLAVINSKI
Project Engineer

APPROVED: Alan R. Barnum

ALAN R. BARNUM
Assistant Chief
Information Sciences Division



FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: RADC-TR-76-329, Vol II (of eight)

2. GOVT ACCESSION NO.:

3. RECIPIENT'S CATALOG NUMBER: (9)

4. TITLE (and Subtitle): SOFTWARE DATA COLLECTION STUDY. Volume II. An Analysis of Software Data Collection Problems and Current Capabilities.

5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report. June 1975 — June 1976

6. PERFORMING ORG. REPORT NUMBER: (14) SDC TM-5542/002/01

7. AUTHOR(s): (10) N. E. Willmorth, M. C. Finfer, M. P. Templeton

8. CONTRACT OR GRANT NUMBER(s): (15) F30602-75-C-0248

9. PERFORMING ORGANIZATION NAME AND ADDRESS: System Development Corporation, 2500 Colorado Ave, Santa Monica CA 90406

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 63728F, 55500810

11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (ISIS), Griffiss AFB NY 13441

12. REPORT DATE: (11) December 1976

13. NUMBER OF PAGES: 118

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same

(12) 126p. (16) 5550

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report): Approved for public release; distribution unlimited.

(17) 08 (18) RADC (19) TR-76-329-Vol-2

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report): Same

18. SUPPLEMENTARY NOTES: RADC Project Engineer: Richard T. Slavinski (ISIS)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number): Software Development; Data Collection Problems; Data Collection Practices; Project Monitors

20. ABSTRACT (Continue on reverse side if necessary and identify by block number):

The current state of software data collection leaves much to be desired. There is no comprehensive base of software development information, and the data that are gathered lack reliability, accuracy and validity. Data gathered from diverse sources and projects are not comparable due to a lack of standardization in meaning, measuring techniques and collection procedures. One of the chief obstacles to gathering more reliable data are project resistance to managerial control and a reluctance to release data that might reflect on project efficiency or reveal proprietary technology. (cont'd)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

339 900

DMW

20. ABSTRACT (cont'd)

This report reviews the literature dealing with software data collection problems, evaluates current military data collection practices, and surveys briefly some project monitor systems that could aid in the collection of more valid and reliable data. The alternatives available in managing a software data collection operation facility are also examined. Actions leading to objective, reliable and standard measures and for removing some of the subjectivity of measures are considered. No easy or fine solutions are seen, but improvements are attainable by technological and managerial changes and through economic inducements to provide better data.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont'd)

1. STATEMENT OF THE PROBLEM

If studies of software quality and software development productivity are to be done, accurate, precise, and valid measures are required. To date, most studies of software productivity, including the definitive SDC (1) and PRC (2) studies, have been based on subjective, after-the-fact estimates of what happened rather than direct recording as events accurred. Attempts to improve software data collection are faced with several serious problems:

- Increased instrumentation of the software development process entails additional costs
- Developers resist the imposition of management controls and reporting requirements
- Mensuration is still plagued by many sources of error variances
- Technological inadequacies exist in both software development and software data collection
- Measures taken for project *management* purposes are not always suitable for methodological comparisons

Studies of the software development process are also inhibited by two other phenomena that are generic to experimental work:

- Heisenberg Effects where the act of measuring the behavior of a phenomenon changes the behavior.
- Hawthorne Effects where the knowledge that they are part of an experiment increases the motivation and productivity of those involved in the experiment

Further, where investigations depend upon a sample of projects, the lack of standardized instrumentation destroys the comparability of measures taken from the projects in that:

- The same measures are not collected.
- Measures are not defined in exactly the same way
- Measures are not acquired or derived in the same fashion.
- Probes are not inserted at the same points in the process.
- Projects are configured differently, inhibiting the collection of comparable measures or the insertion of probes in comparable points.

The inability to collect valid data to predict software productivity and quality may be a serious hinderance to the efforts now underway to improve software productivity.* Even where a methodological comparison shows some advantage for a particular technique, huge interpersonal and quotidian variances cast doubt on the results. For instance, in the SDC studies (110 factors, 169 projects) the best predictive regression had a standard deviation greater than its mean. The confidence that one can place on such results is low indeed.

In view of these considerations, the objectives of this investigation were to (a) investigate the sorts of software data collection problems encountered, (b) derive an estimate of their seriousness, (c) suggest some therapeutic measure to ameliorate their seriousness and (d) evaluate current software data collection practices in light of these considerations. Additionally, the question of the most efficient and effective way of managing the operations of the data collection system was also examined.

*Software takes a rapdily growing proportion of EDP costs, estimated at 80% of costs by 1980. Programming productivity has been increasing very slowly as compared to computing power, but just as serious as overall costs is the high risk nature of software projects whose slippage may cause even more serious delay in high priority defense, space or commercial systems.

2. SOFTWARE DATA COLLECTION PROBLEMS

Although the literature tends to support the precis of the problems given in Section 1, no studies were found that directly attacked the evaluation of the seriousness of the problems or of the efficacy of solutions to them. The SDC Studies, the PRC Studies and the Huntsville Repository [50] all deplore the subjectivity of their data, much of which depends upon fallible human recall, and call for more objectivity. These sources cite the uncertain nature of developemental activities, the insubstantial nature of software and the lack of understanding of the software development process as sources of (apparently) poor estimates of developmental time and costs. Despite considerable effort on the part of DOD agencies to establish standards for software development, the lack of standard terminology, software development models, measures and methods of mensuration is still deplored. Although the lack of quantitative data may not entirely invalidate the identification of problems, it casts doubt on their degree of seriousness, makes predictions tenuous, and renders research results subject to question. While further research may eventually quantize these problems, the current lack of knowledge limits the recommendations that can be made concerning solutions to them.

The review of the literature and our discussions with project and repository managers revealed several general classes of problems related to:

Mensuration Techniques
Instrumentation effects
Reliability of measures
Resistance to management control
Costliness of data collection
Control system operations

2.1 PROBLEMS OF MEASUREMENT

A basic problem for software data collection is determing how closely to monitor the software development process to enable both project management and methodological research to be done. If the information loop is closed, mensuration involves not only determining what information to be gathered and the weight to assign to it (mensuration) but the selection of sampling points and the insertion of probes into the process (instrumentation). This also involves evaluation of the measures to determine what information they yield concerning the present operation and what implications may be drawn in general and about future impacts (interpretation), and the determination of the proper responses to take to correct, improve or reinforce system behavior (reaction) is then of concern.

2.1.1 Mensuration

The number of measures that could be taken is potentially quite large. The SDC Studies (1) eventually derived 152 indices and that list is far from exhaustive. For instance, the study contained few measures of software quality, a factor that Weinberg (4) has shown experimentally to have profound effects upon programmer "productivity". The selection of the proper measures to take and the weights to be assigned to them is not necessarily an easy decision. Very few of the indices used in the SDC Studies had predictive value for the productivity criterion used-lines of code per man month. However, such a criterion may have little validity for some aspects of software development, system analysis or user manual preparation, for instance.

For project management, measurements are usually derived by:

a. Establishing a performance standard (a schedule, a budget, or a product design)

b. Collecting reports of work progress, resource expenditures, and system content

c. Determining how actual performance varies from system content

d. Rating the seriousness of measured deviations

To establish standards and to interpret the meaning of the measures, much needs to be known about the characteristics of the project and its environment and how these relate to the body of past experience and knowledge about software projects.

Although the process for deriving measures seems clear and quite a lot is known about software development, by and large we really don't know the critical indices of quality and progress nor the weight or importance to assign to them. The importance of quality is heavily influenced by project objectives and environmental characteristics often determine levels of performance to expect so that the same importance cannot always be assigned to a particular delay or deviation. Sometimes a measure is only meaningful in context or must be derived by comparing a mass of detailed data against a complex criterion of excellence.

Many times software development measures are seemingly taken just because they are available, not because a real need for the measure has been established. At our present state of knowledge, this may not be surprising. Indeed, in a research repository environment where it may not be possible to anticipate all the questions that might be asked, it is difficult to assess the relevance of any measure or to know that all appropriate measures have been defined.

The choice facing data collection ranges from all possible data with the risk that much of it may never be used (and may be useless) or only that data pertinent to project objectives (including methodological experimentation, of course.) In consideration of the relative sparseness of software projects in any given class and the overall duration of projects, it would seem that the shotgun approach would be superior because years may elapse before an answer

would be available from the sharpshooter approach. There is a risk that without adequate analysis and evaluation of research objectives even the shotgun approach may not yield pertinent information. Actually, the optimal approach would seem to be to define the minimum subset of measures necessary to answer ordinary productivity and quality evaluations with the option to add specific measures to answer unforeseen or unusual research questions.

2.1.2 Instrumentation

The selection of the points in the developmental process at which to gather information, the frequency with which to sample performance, the depth of detail required, and the means used to gather the information are the major decisions to be made concerning instrumentation. Each delivery of a product provides an opportunity to evaluate the quality of the product and of the process that produced it. Hence, if the process is to be monitored closely, numerous intermediate milestones and products should be identified. If monitoring is lax, only a few or no intermediate products need be required.

Sampling frequency is usually accompanied by variations in the organizational depth and detail in which measures are taken. These factors determine the granularity of the control that can be exercised. If products and progress are measured only at long intervals, at the top level of management only, and only general estimates are sought, the granularity of control is coarse. If frequent progress reports, small increments of development, individual worker reports in great detail are sought, granularity may be fine.

There is a risk associated with coarseness and a cost associated with fineness of control. Sampling progress too infrequently and too generally permits large variances in schedule delays, cost overruns and inappropriate design to build up without an opportunity to take corrective action. Taking such risks seem justified only when the onus of the risk-cost rests upon the supplier and not upon the procurer. If payment is not made until a satisfactory delivery has

been made, the procurer may lose nothing but time but may be denied the use of a very important product. For large systems, this is most unsatisfactory for both developer and procurer. The developer cannot normally afford to invest large sums in salaries, facilities and equipment that will not be reimbursed for years and the procurer cannot afford the risk of receiving an unsatisfactory product. Pragmatically, of course, the procurer may not be able to reject the product no matter how substandard it is - much software has been accepted on just this basis. Cost-reimbursable, incremental and frequently reviewed contracts reduce the cost risk to the supplier and the risk of an unsatisfactory product to the procurer.

On the other hand, too frequent sampling is expensive. Preparing progress reports and intermediate documentation and performing frequent reviews do take time and money. If the procurer (or developer) insists that intermediate products be approved before risking further investment, opportunities arise for extensive delays. If the supplier proceeds with development prior to approvals, he risks many manhours on a product that may be unsatisfactory. (This "premature coding" before design review has also resulted in delivery of shoddy merchandise.) In the military, the elaborate configuration management practices (a la AFSCM 375-7) that have been developed to combat the ills resulting from lack of control are often applied religiously regardless of project size. In short, a better-safe-than-sorry attitude and much opportunity to hide the costs of a massive control system within the military bureaucracy has also produced some very costly software of very little better quality than under more lax supervision.

The literature is full of calls for greater management control of the developmental process [8, 9, 10, 11, 12, 13, 14, 15], but the evidence in support of the assertion is tenuous. Gerloff [16], for instance, after a study of 108 R&D contracts (mostly non-software) concluded that the application of a large volume and variety of management control techniques to R&D contracts

could not be associated with decreased technical, schedule or budget difficulties. In fact, projects with less control reported fewer failures and difficulties*, but this may be an artifact of measuring.

Although the trade-off between fine and coarse granularity is obvious, the shape of the function has not been established. Risk is a function of the urgency of the need, the rigidity of schedules and budgets in relation tc the availability of manpower and facilities, the seriousness of meeting performance criteria, and the degree of innovational uncertainty in the project. Granularity of control should be fitted to the risk, but only a general estimate of the trade can be made. This, too, is a prime area to be addressed by repository studies.

The method of measurement is also a momentous problem for instrumentation. Many if not most software development measures currently collected are highly subjective. Many are not even observational, but matters of opinion. Design reviews, for instance, seldom have precise review criteria established for them and the personal bias and unique professional experience of the reviewers often greatly influence the critiques. Observational and direct recording techniques lead to finer granularity and greater volume. Automated evaluation tools and instrumented support tools produce an even greater amount of data. These data are more objective than subjective estimates of progress and the actuarial accounting for resource expenditures, but the sheer volume may tend to confuse and overwhelm project monitor personnel. Hence, some reduction of data by sampling, summary or interpretation may be necessary if the data are to be useful to human controllers. The volume of data may also be too large to save for research work unless processed. It may be best to leave such fine granularity to data collected for specific studies. The objectivity is to be highly prized, but perhaps it can be preserved in a summary form.

*An associate of the author who conducted a series of supervisory techniques training experiments in the early '50's remarked, "There is very little that a supervisor can do to spur production, but a hell of a lot of things he can do to hinder it". This may fit many of our managerial control attempts.

2.1.3 Interpretation

Knowing what interpretation to place upon data is the key to both managerial and research results. Although the interpretations made are usually subjective (and hence subject to a certain amount of bias and unreliability), statistical techniques have been used to determine the predictive value of some measures. The SDC Studies, for instance, used multiple regression techniques to predict costs and progress, but not product quality. Ellingson [47] and Tucker [48] did do so, however, with good success. Other attempts have not been so successful. Sachman [6] found interprogrammer variation so great that, with a small sample of people, it overshadowed the conclusions that interactive was superior to batch-oriented programming. Similarly, Reinstedt [7] found no reliable personnel selection predictors of success as a programmer.

Part of the difficulty with statistical prediction may be that measurement exists in a dynamic environment. If a difficulty or variance arises, the manager immediately takes action to correct the difficulty. If successful, there are no longer any deletorious effects to predict. This colors any attempt to make statistical predictions. In short, the unreliable nature of the data, the difficulty of performing experiments tc verify interpretations, and the general low level of present knowledge hinder accurate interpretation of the implications of the data that are collected.

2.1.4 Reaction

For management, reactions close the loop in the process control system; for research, they are the means to improve the developmental process. Just as the proper interpretations to place upon collected data are obscure, so are the proper reactions. Much of the research that may be done with the Repository rests with establishing the implications of measures for productivity and product quality and determining the effects of various reactions to these.

2.2 INSTRUMENTATION EFFECTS

The very act of instrumenting a process can influence the behavior of the process observed, thus destroying the comparability of the project to other such projects. These effects are generally classed as 'Heisenberg Effects' and the subset of these known as 'Hawthorne Effects

2.2.1 Heisenberg Effects

In general, Heisenberg Effects are any changes introduced into a phenomenon by the act of measuring or observing that phenonmenon. Inserting probes into a process may cause time delays in the process while the measure is obtained. It may disturb or distract the processor, causing it to make errors or otherwise malfunction. It may change the behavior of the processor or of the phenomenon observed in some way.

Development personnel frequently complain about the amount of time to complete project report forms and write progress reports. (It should be noted that they also actively resist attempts to make such data gathering more automatic--time punch cards, for instance.) Even placing probes in programming tools and operating systems to record data introduces some time delay. To the degree that these activities cause significant delays in development or in computer turnaround on jobs, project performance is degraded. It should be noted that significant amounts of project time can be devoted to preparing for major project audits - preparing presentations, making audio-visuals, dry running presentations, and conferencing.

Equally disturbing may be the distraction effects of having to stop concentrated technical activity to write progress reports and prepare briefings. Even a minor interruption may cause one to lose the train of thought and it may be difficult to pick up the threads again after any considerable interruption. On the other hand, too much emphasis could easily be placed on minor interruptions. There are many such interruptions during the normal working day, both social and technical, that do not interfere with the job. No experimental

evidence in the data processing field exists concerning the impact of intervening activities, but psychological learning experiments show that immediate recall of information is affected by intervening activities. (As a graduate student, the author participated in an experiment on learning lists of nonsense syllables. A rest period with some socializing interferred least; an entirely different activity (solving a mechanical puzzle) was intermediate; and a very similar activity (learning another list) interfered most.) The length of the intervening period of activity is undoubtedly important since fine detail is rapidly forgotten with time. Hence, a report (like a cost log) that takes only a few minutes to prepare has little impact. A written progress report that may take an hour or so to prepare has a noticeable effect and a report or briefing that takes a day or more to prepare may be quite damaging.

There is also the emotional impacts of the tasks. Technical personnel dislike making administrative reports and view them as non-productive. Others find written compositions difficult and distasteful. If active resistance to writing progress reports does not develop, considerable frustration, irritation and emotional turmoil may. The emotional upset may have as much or more impact than the intervening activity and the temporal interval.

Even though the technical person may not actively participate in the data gathering activity, the fact that his performance is being closely observed may change his behavior. He may proceed more cautiously and meticulously then he otherwise might. Also, he might resent being observed so closely and try to evade such scrutiny. Or, close scrutiny might make him nervous and cause more errors to be made. Other impacts of observation are discussed in the next section.

2.2.2 Hawthorne Effects

Briefly, Hawthorne Effects are the extra motivational effects created by the performer knowing that he is participating in an experiment or that his work is receiving special attention. The extra motivation causes the person to work harder, pay more attention, and be more interested in his work. These effect were first noted in a series of experiments of working conditions in the Hawthorne Works of General Electric between 1927 and 1932. In these experiments, almost any change (or none at all except the observational act) resulted in an improvement in performance. The same sort of phenomenon has been often noted by time and motion study men in taking work samples to establish standards. Almost invariably, unless the employee deliberately stalls, performance during the criterion trials is better than the average performance of the workman on an ongoing basis. These early studies led to a large volume of motivational research with the general conclusion that almost anything that causes the employee to feel that his job and he are important will result in improved performance.

The possibility of Hawthorne Effects immediately calls into questions any experiment in which experimental results are compared to "industry norms." Such is the impact of motivation on productivity that any methodology can be shown to be better than the norm just by telling the performer he is in an experiment and by gathering data on his performance.

2.3 UNRELIABILITY OF MEASURES

Reliable measures - measures that mean the same thing every time they are taken and that accurately reflect the attribute that they purport to measure - are necessary for valid comparisons of projects or methodologies. Among the many factors that contribute to the unreliability of measures are the lack of standardization and the intangibility of the software development product and software development measures.

2.3.1 Standardization

In part, the lack of standardization in data processing is one of terminology - the same term is used to mean several things, the same thing is referenced by different names. Standardization's other aspect is real - standard measures do not exist for some important concepts and the software development process measured is not configured or performed in the same way.

The level of standardization varies considerably over all aspects of data processing. The EDP Analyzer for August, 1975 [23] delineated "The Benefits of Standard Practice" as requiring standards for:

- Documentation (Specifications, Descriptions, Manuals)
- Project Management (Project Structure, Project Planning and Control, Progress Reporting, etc.)
- Data Standards and Standard Data Definitions
- Programming Standards (Design, Coding, Testing)
- Integrity Standards (Restart, Recovery, Backup, Control, Security)
- Hardware Standards
- Manual Procedures Design
- System Software Standards
- Turnover Standards
- Computer Operations Standards
- Change Implementation Standards

For most military systems, to this list should be added:

- System Analysis Standards
- Configuration Management Standards

Obviously, having some standards is not important for all projects - i.e., integrity standards might be easily foregone for a one-shot program and turnover standards may not be important for research projects except insofar as applied to the final research report. Equally obvious are the benefits of standardization for data collection and analysis. For data to be comparable, a named measure must mean the same each time it is collected. As evidenced in the recent symposium on software reliability [24] where several definitions of "reliability" were given, people do not always mean the same thing by a particular term. Although standard terminology may be employed, measures taken at different times and under different conditions may be influenced by the varying factors and be equally non-comparable.

In one of the SDC Studies, Weinwurm and Zagorski (1, TM-2712] deplore the lack of data standardization, characterizing software systems as intangible, unique one-of-kind, custom-made products produced without product or work standards. They cite little agreement on concepts and terminology and poor definition of measures and system attributes and an insufficient understanding of the developmental process as factors in the unreliability of their analytic results. From another viewpoint, they point out that much of their data was gathered for legal cost accounting purposes rather than for research and analysis. Unless the research goals are explicitly stated with the data that will be taken as evidence of proof or disproof of a hypothesis, ambiguities are likely to remain.

Similar comments on standardization are made in the IBM Structured Programming Methodology Report [25]. The report cites the difficulties encountered in measurments of the 152 variables in the SDC studies and advances definitions of project phases, reporting levels, repository functions, and data classes and types. The definitions advanced are in accord with current data processing concepts but are not as formal as might be desired.

In 1970, Weinwurm again summarized the situation in software development data collection [26], calling it "transparently superficial". There is little hard data; most evaluations rest on qualitative and subjective considerations. It is difficult to establish standards in the face of rapid technological obsolescence of the measures that do exist, but performance (productivity) measures are needed to establish baselines for cost estimation and assessments of project performance. Schlight [27] at an AFSCF workshop in Aeronautical Systems Software makes a strong argument for a common vocabulary for systems work in overcoming interdisciplinary differences. The systems engineer, the system analyst, the programmer, the quality assurance staff, the project manager and the customer are each experts in his own portion of the system development activity but interfaces fail because of the lack of a common vocabulary. Common measures are only part of this vocabulary, but misunderstandings exist even on that level. At the same Workshop, Wolverton [28] set some goals for standards, ranging from establishing software as a real-world entity through giving it proper visibility to setting standards for the expression of algorithms. Standards may also establish the adequacy of user/buyer requirements definition and a proper balance of costs vs performance vs availability in software development.

In a somewhat different vein, the Janaury 1975 issue of the EDP Performance Review [29] was devoted to standard metrices for software performance. The metrices advocated cover the efficiency, effectiveness, reliability and throughput of a system. The benefits to be realized are improved communication among users and developers, better guidelines for new personnel,

reduction of duplicate effort, more complete and consistent cost accounting, and more consistency of system outputs. To these we may add data adequate for comparative studies.

There has recently been several strong moves to improve the standardization of program code, notably the structured programming call for using only three control structures (concatenation, repetition, selection) (Hansen [30]) and for using good programming style [31, 32] to increase the readability and ease of understanding.

There are really two aspects to standards: establishing them and enforcing them. Mathis and Willmorth [33] consider at length some of the impacts of permitting exceptions to documentation standards or not enforcing them. Their conclusions seem to be that 'standards' are normally minimal levels of acceptable practice necessary to adequate performance. Documentation variance and deficient review procedures and criteria are usually indicators that information is not available and that confusion, production delays and errors will result while the information is generated. Acting upon inadequate and missing information leads to many erroneous design decisions, operating inefficiencies and expensive error correction and retrofitting.

2.3.2 Subjectivity

Although all of those who deplore the lack of standard measures cite the subjectivity of measures, the intangibility of the product and the process, and the lack of understanding, it would appear that much of the subjectivity results from the failure to utilize the information that we do have or to generate the information that is required. Fred Brooks [5] speaking about schedule estimating techniques says (a) our techniques for estimating schedules are poorly developed, being tacitly based on the assumption that all will go well and (b) since estimates are uncertain, software managers are loath to defend them. These factors plus the pressure management and

the client exert toward keeping costs down, very often result in grossly underestimated schedules. Brooks further feels that management often assumes a linear trade-off between time and manpower - that is, that estimating techniques confuse effort with progress. Schedule progress is poorly monitored and software development has been traditionally reluctant to adopt practices that are standard in other disciplines for doing so. That developers are most persistently optimistic in schedule estimates is shown by a study that Brooks reports. Schedule estimates were made at two week intervals throughout a project. In general, no estimates of schedule duration were changed before the activity started. During the activity over-estimates were steadily shortened, but underestimates did not change until the missed completion date was emminent. In part, Brooks feels that this is because no one likes to be a bearer of bad news. Although perhaps less lethally than in the good old days, heads can still roll if the chieftain is displeased. Even if not fearful of the reaction, many people are reluctant to cause pain. Even when the chance of failure is evident, people continue to hope for the best, expecting a miracle to save them. Others just "grin and bear it" rather than baring their troubles to others or actively seeking alternatives.

Contributing to these psychological problems is a lack of understanding of the process and of the product under development. Development has always been high risk endeavor but software systems (and other high technology developments) are often treated as if no uncertainties are left unresolved. Familiarity with the application area, familiarity with the customer, and familiarity with the solution algorithms greatly enchances the changes of successful development. Definite learning curves have been demonstrated in producing and cost estimating, for instance, as Schwartz [34] shows for successive versions of compilers and Wolverton [35] in producing the second version of a system.

A common failing in cost estimation is giving inadequate attention to support activities. On military contracts, documentation is normally more voluminous than on less controlled contracts and its cost is often underestimated. Decision delays in reviews and the revisions to documents that follow may not be accurately estimated. Hartwick [36] cites support tools, requirements compilation and validation and verification as areas often given inadequate consideration. He also accuses developers of inadequate analysis of the job to enable full understanding. Techniques do exist to break a development task into all its component elements and even to perform sample design, coding and simulation to gain greater understanding. Such analytic activity does take time and effort, but may not be as costly as making a number of poor projections. Expert consultation (people who are familiar with the application, the customer or programming techniques) can help.

A number of these considerations, summarized in Table 1, were presented by Keider [37] in an article on "Why Projects Fail". A somewhat different list is presented by Mathis and Willmorth [33] in Table 2.

Thus, much of the subjectivity and unreliability of estimates of cost, schedule and product characteristics are alleged due to mismanagement rather than inherent intangibility or lack of knowledge. Joseph [38] agrees on the actions to take - building up a statistical base of project details and time spent, using previous similar projects for comparison and special pricing of differences, and estimating crucial factors - but warns managers to avoid "sludge" factors - adding time to cover anticipated slippages. Activities will soon expand to take up all the allowed time.

Of all the tools in the system development box, the one that seeks to take advantage or compensate for the subjectivity of estimates is PERT (network scheduling). There, three estimates of activity duration are made and weighted in computing an average duration. Various levels of probability of completing early or late can be computed assuming the estimates reflect

Table 1. Factors in Project Failure
Source: S.P. Keider, Why Projects Fail, Datamation, Dec. 1974

A. Pre-Initiation Period

1. No standards for estimating project length
2. Estimates not made by the project leader or other persons responsible for performing to the estimates
3. Project inadequately defined
4. The short lead times allowed for estimations result in corresponding inaccuracies
5. Personnel availabilities for the project were unknown
6. Staff desires regarding the project were unknown

B. Initiation Period

1. Little or inadequate documentation available on interfacing systems or the existing system
2. Project leader responsibilities undefined
3. Paper flow poorly defined (Decision mechanisms - responsibilities, acceptance criteria, system objectives - ill defined.)
4. Inadequate knowledge of design 'tools'
5. Project definition vague, misleading or erroneous
6. Changes in requirements not reflected in changed estimates
7. Inadequate or no time spent in planning
8. Problem avoidance techniques not understood or considered
9. Resource requirements (acquisitions by number and type) not scheduled
10. Program team activities not clearly presented to user/customer
11. Project leader not told what constitutes project completion

C. Duration of Project

1. Project information not posted or reported
2. Project reviews and audits trivial
3. Personnel changes made without considering (or reflecting) the effects on scheduling
4. Adherence to (and enforcement of) standards and specifications lax
5. Unanticipated resource requirements (e.g., for data entry, computer test runs, or design reviews) arise
6. No use of previous project experiences, tools or work simplification methods
7. Project lacked a manager for all or part of the time
8. A project log or file not kept
9. No project audit trail maintained

Table 1. Factors in Project Failure (cont'd)

C. 10. No project milestones set
11. Staff members considered equally expert in all functions, i.e., no attempt to take advantage of special knowledge or skills of individual task members.
12. Company philosophy demands the "maximum utilization of personnel" i.e., organization of work was not project-oriented.

D. Termination of Project

1. Project history and statistics not determined or updated
2. Quality control measures not used
3. Knowledge gained not saved or transferable
4. Personnel are not evaluated
5. Project not formally turned over or terminated
6. Recommendations for enhancement not documented

E. Post-Termination

1. User satisfaction not determined
2. No evaluation of results vis a vis objectives
3. Integrity of data not maintained
4. Areas of relative freedom from problems not evaluated
5. Quantification of changes to requirements not done
6. Usefulness of information not evaluated.

Table 2. Causes of Schedule Slippage, Excessive Costs, and Substandard Products on Software Development Projects.

Cause	Symptom
Failure to manage	• Vague (or no) management plan • Vague (or ineffectual) organization • Poor communications • Difficulty in getting decisions made
Failure to understand the problem	• Vague statement of requirements • Vague statement of operational environment • Vague quality assurance provisions • Inadequate assessment of risk
Failure to acquire needed capabilities on time	• Insufficient personnel • Insufficient computing capacity • Insufficient program production tools
Failure to provide adequate communication	• Responsibilities and authorities not clearly established • Project contact points not established at a working level • Project procedures not established • Libraries not established
Failure to maintain project ability	• Management turnover • Key and technical personnel turnover • High frequency of design change • Rapid changes to work assignments
Failure to provide adequate quality assurance program	• Programming standards and conventions • System and software acceptance specifications • Precise performance requirements • Independent integration and test programs • Review criteria • Published test plans, procedures, and results • Change control procedures and rapid response to reported discrepancies

SOURCE: Mathis and Willmorth, Software Milestone Measurement Study

the actual situation. That these estimates are inaccurate was shown by King and Wilson [39]. They found that schedules tended to underestimate actual performance by about 30% before the activity began and by 21% during the activity. They give a formula for adjusting estimates upward (roughly by adding a standard error of estimate to the duration) to compensate for the tendency, but give no evidence to indicate persons met the adjusted schedules any better than the estimated ones.

Corrigan [41] aptly covers many of the factors leading to unreliable and distorted information. First, most information arriving at a project management office has been processed through a cascade of time delays from the individual programmer through several layers of management until the information is no longer timely or current. In the process, the information is also filtered and distorted. In averaging and summing and summarizing, it is distorted toward the norm and toward the optimistic. The impact of learning that the reported information is distorted is deletorious - it demoralizes the individual reporter and discourages him from making accurate reports and elicits angry and punitive reactions (often exaggerated) from the project monitor. Second, there is a lack of feedback from reporting either in terms of evaluations of project performance or evaluations of the excellence and accuracy of reporting. The project monitor does face a dilemma here in that it is very difficult to punish a project for poor performance and at the same time reward it for good reporting. That is, the reward for quickly reporting poor performance is quick retribution, not praise for telling the truth, which tends to discourage accurate and timely reporting.

2.4 RESISTANCE

Organizational resistance to the imposition of management controls and the collection of performance data is an almost universal managerial phenomenon. Among the various studies of the phenomena is one by Thambain and Wilemon [40] in which they polled projects and obtained management ratings of conflict concommitants. The top seven items in the survey - those endorsed by more than 50% of the participants and receiving high ratings by managers are shown in Table 3. It would appear that resistance to data collection efforts is more likely to develop when any ambiguity in any aspect of a project exists - in goals, responsibilities, authority, or interrelationships - or when a threat is seen to exist (especially when the power structure is weak or ambiguous and non-compliance may go unpunished) While the investigators were looking at more than the honesty of reporting, any conflict situation where misunderstanding or disagreement exists on the part of those managed is likely to lead to less than complete cooperation in data collection efforts.

The personal feelings of project participants that may lead them to resist data collection efforts seem to be:

a. A reluctance to be a bearer of bad new

b. That one's privacy is being invaded

c. Antagonism to a perceived threat

d. A reluctance to change one's normal mode of conduct

e. That the effort demanded is excessive (too costly, too interfering, too effortful for the perceived benefits)

Table 3. Conflict Factors in Management Control

SOURCE: Thambian Diagnosing Conflict Determinants in Project Management, IEEE Trans Eng. Mgt.

PROPOSITION: Conflict is more likely to develop between project management and project membership when:

PCT POLL MGMT	PROJ MGMT RATG*	
98	4.8	1. The less the specific objectives of the project are understood by the project team.
95	4.5	2. The more the members of a functional area perceive that the implementation of a project system will adversely affect their traditional roles and responsibilities.
90	4.5	3. The greater the ambiguity of roles among the participants of a project team.
80	4.2	4. The less the agreement on top management goals.
75	3.8	5. The lower the project manager's formal authority over supporting organizational units.
70	3.5	6. The lower the project manager's power to reward or punish supporting organizational units and personnel.
50	3.1	7. The greater the diversity of disciplinary expertise among the participants of a project team.

*A rating of 3 was the indifference point on a six point agree - disagree scale.

2.4.1 Bearers of Bad News

The reluctance to give pain whether from compassion or fear of retaliation was discussed above. The reporter may defer revelation or play down a difficulty both through optimism - a belief that the difficulty will be overcome before it really adversely affects project performance - and through reluctance to lose stature in the eyes of a valued or feared person and a hope that one can recover from the situation before the degradation is necessary. Of course, once the revelation is inescapable or the difficulty uncovered through other sources, the impact of the situation is made much worse by the attempts to coverup.

2.4.2 Invasions of Privacy

Many developmental and managerial matters are viewed as only of internal concern by corporations and individuals alike. That is, it is believed that management control systems should be functionally "black box" systems. Monitors of projects should be aware only of the stimuli (inputs), the control rules and the responses (outputs) of the software development system and not concern themselves with what goes on inside the box in terms of understanding the precise functioning of the process. Besides a general none-of-your-business" attitude, some justification for this feeling may exist for corporations in preventing the disclosure of "proprietary" techniques and information that are thought to give a competitative advantage over other companies and for individuals in preserving the mystique of the art or profession. That is, special techniques and knowledge are what the corporation or individual has to sell and giving them away represents a very positive debit to the revealer.

2.4.3 Threat Reactions

Management control systems often enforce performance standards by the imposition of penalities, the granting of rewards, and/or the threat or promise thereof. The reaction may be positive - conformance and increased effort - or negative. Negative reactions that lead to unreliability in the collected data include defiance of the restrictions, and attempts to evade punishment by coverup, falsification, withholding information, sabotage and other counter-aggressive behavior. Wolverton [28] cites reluctance to supply software reliability measurement data due to participants' regarding such data as measures of their proficiency and ability. He also reports that it is unpopular to contract for "failure - analysis" both because it seems to be a self-fulfilling prophecy and because it is a tacit admission of fallibility.

McGregor [42] discusses management control systems at some length. Among the problems he cites are:

- Widespread antagonism to the controls and those who administer them
- Successful resistance or non-compliance by many employees at all levels up to the top
- Unreliable performance information because of employee antagonism and resistance to administrative controls
- The need for close surveillance of employees dilutes delegation of authority, cuts into managerial time and impedes the development of workers
- High administrative costs accompany stringent controls

McGregor questions many instances of the imposition of standards in that the apparent goals are too high, that perfection is demanded, that there be no errors, no turnover and no time off. What is a reasonable goal? If expectations are too high, attaining the goal is not regarded as possible and the standard will not be practiced or enforced. However, much of the resistance is based on the perception of the standard and not on an objective evaluation.

One difficulty in this situation is that the rewards for compliance are long term and uncertain whereas the punishment for non-compliance is immediate and punitive. Hence, the individual's chief goal is to escape punishment, not to do a good job. In keeping with the rule, "Non-compliance tends to appear in the presence of a perceived threat", the resulting behavior is defensive, protective, resistive and aggressive. The individual tends to perceive "accountability" as "find out who goofed", not as an attempt to solve problems.

The actions taken to demands to change or for more information range from failure to respond to attempts to "beat the system" to active dishonesty. The simple failure to comply or respond is evidence by those who "forget" or repress the memory of the request and those who ignore the request, delay response or respond grudgingly. Often, in the case of a preceived threat, human ingenuity is exercised to defeat the purpose of the control system. If the system can be shown to be excessively costly or distracting or misleading it can be disposed of without retaliation. Further, threat becomes a justification and rationalization for dishonest behavior. This is reinforced by an active temptation to cheat since there are rewards for successful lies. In short, McGregor concludes that conventional management control systems have a strong tendency to generate and accentuate the very behavior they seek to prevent: non-compliance.

The therapy that he suggests for this condition is one that project-oriented organizations are alleged to foster: all-out commitment to project goals. If individuals are committed to goals, if they have a sense of mission, they will stand against great deprivation, work long hours in uncomfortable conditions and make many personal sacrifices for the job. However, if management implies it does not trust the individual or uses threates to elicit desired response, if it drives instead of leads, loss of mission-orientation and resistant behavior quickly results (It should be noted, however, that sometimes projects develop the team spirit to the point where they are bent on succeeding despite the oppressive behavior of management - a situation with both good and bad results.)

McGregor's strategy for including commitment is to have an open presentation and discussion of management's view of the requirements for successful completion of the project, including the "restraining forces". There should be an analytic presentation of the required performance, including any changes to standard or normal or past procedures. There should be a clear analysis of the contributions made by individuals and subunits and of the importance of these contributions toward projects goals. The participation of individuals and subunits in setting goals and standards must be encouraged and suggestions heeded. It will be noted that usually the project will set higher goals and be a severer critic than management would care to do. Without this commitment, there is little likelihood that reliable data can be acquired; with it, the project is proud to present achievements and actively looks for problems to solve rather than avoiding them.

Fleck and Hodge [43] in discussing user resistance to automation also cite overt opposition, loss of morale, withholding of information, reporting of inaccurate or partial information and actively destructive, obstructive and discreditive behavior. They, too, stress openess in installing procedures, including the communication of reasons, the soliciting of active participation, and thorough training and indoctrination of persons in the new procedures.

Ryle [44] points out that technical people have been so resistant to the imposition of configuration management rules that configuration management becomes, by default, the responsibility of accountants and administrative personnel. Unless configuration management can be presented or applied as a technical activity - i.e., as part of the specification and implementation process - it will fail.

2.5 COST FACTORS

Almost every attempt to install more exacting system life cycle measurement procedures is met with the argument that it will cost too much. This allegation may be true in part, but it could also be a resistance ploy by the individuals and companies involved. At SDC, formal cost collection and project reporting is normally estimated at 3% of project costs, a figure that agrees very closely with those reported by Wolverton [28] for "Program Control." On the other hand, these formal estimates may ignore many informal costs and time lost on a working level. They certainly do not contain the collecting of detailed error and productivity statistics. For the 427M contract actual charges from the program data office for Program Control (schedules and budget reports) was 1.02%, for Configuration Management (change processing) was .56% and for Quality Assurance (review and discrepancy processing) was .90% of total costs. Together they approximate the 3% estimate, but hidden costs could double this amount.

An evaluation of a number of life cycle models for estimating developmental costs (and other statistics) was made by Walker [45]. Although there were a number of reasonably realistic simulation models available, the models were seldom if ever used. The main reasons given for not doing so were the lack of valid data and the costs of collecting and maintaining the data base for performing the life cycle analyses. Tied to this is the fact that it takes a long, drawn-out effort to collect the data, an effort that seems not justified in the face of the questionable conclusions of the simulation

models. In support of this conclusion, Walker cites as an example an attempt by a military service (unspecifed) that tried to establish such a data base and use the life cycle models to predict costs and performance characteristics. The effort was abandoned due to the high cost of maintenance, unmanageable input, and lack of application. Although the effort was deemed not economically practical and of limited benefit, it should be noted that system costs and especially software costs have been greatly accelerated since this study was made. Interest is at a much higher level and the potential benefits may be much greater. Data base and data collection technology has also advanced considerably since the 1960's so that collection and maintenance of the data base ought to be technically and economically easier to manage.

The major factors of cost in software data collection are:

- Data granularity
- Data automation
- Production goals
- Indirect costs

2.5.1 Granularity

Increasing the number and variety of measures taken or increasing the sampling frequency of data collection will increase costs. It is also likely to improve the accuracy and reliability of the data and may extend mensuration to cover previously unmeasured attributes. Both of these could be valuable in improving the effectiveness of management control and providing increased insight into the software development process. Finer data may be a prerequisite to making valid comparisons of programming methodology.

From other discussions it would appear that current measurement practices are deficient in both the early, requirements analysis and operational system design and the late, operation and maintenance phases of software system development. Discrepancy and modificatin histories are fairly easily obtained during the maintenance phase, but normally are collected for only a small proportion of the possible systems. Problems arise frequently in early development phases but are normally treated as part of the system analysis task and seldom recorded either as discrepancies or modifications. Requirements specification was rated by the Government/Industry Software Workshop [46] as perhaps the most important document in the software life cycle and one that is most often thought to be unsatisfactory. Without these specifications accurate costing and adequate designing cannot be done nor can testable criteria for the software be established. These views are in accord with those expressed by SDC in its standard software development procedures -- which also states that the software development plan (almost pure software development data) is perhaps the second most important document produced for the system.

The data that is most accessible tends to be the data that is collected and these are data from the program implementation and test phases. For instance, program module size or subsystem size is much easier to measure than the size of an operational function. Since lines of code produced is an easily measured parameter, it tends to get used as the sole measure of productivity. Errors detected during integration and test are much more public than those encountered during design and coding and tend to be reported more often.

In some instances, more detailed data seems desirable for research than for management control. In error processing, about all management wants to know is that an alleged error was discovered and diagnosed and was corrected. Research wants to classify the errors, perhaps into dozens of categories, and to determine when the error was introduced, what diagnostic techniques were used for detecting and evaluating it, how was it corrected, and how much

the whole process cost. On the other hand, research tends to look at large chunks of the project in determining the costs and elapsed time of project phases whereas management needs an almost day to day and hour to hour fix on these. Much more faithful attention would be given to the data collection tasks if they were separately funded, even though a relatively small proportion of added costs would accrue if additional data collection requirements were levied.

Even though research does not need fine schedule and cost data, more accurate estimates and expenditures can be obtained if the project is decomposed into relatively small tasks and products. Hence, even if the data are summed over a phase for storage in the data base, it would be expedient to collect them in a much finer granularity at the working level.

2.5.2 Data Automation

There are several aspects of data automation that could affect software data collection costs. Among these are:

- Development of evaluation tools to provide very fine data on program statistics.
- Instrumentation of design and programming tools to record run results and product characteristics
- Development of project monitors and standardized local data bases from which data for central storage could be extracted
- Installation of a distributed data entry system for rapid reporting

Each of these steps could result in lowered costs for collecting software data of fine granularity. Each also represents a major investment to develop, install and maintain the feature. In the past, compilers and operating system have been instrumented both in terms of their own operations and the programs they operate on. Such instrumentation delivers large volumes of information, most of which is never inspected. Caution is needed in gathering such fine detail to determine what is needed rather than what is available. Data useful to the individual programmer in evaluating his run may not be pertinent to productivity and reliability research and summary or filtered information might be more useful. IMPACT, for instance, accumulates run time and run results by module and operation, but detail files may be optimally saved.

Similarly for evaluation tools. A program flow analyzer can turn out a great deal of information about the composition and operation of a module and test runs. A code auditor can evaluate program source text for conformance to programming standards. A code verifier can list many errors. This is valuable information for the individual programmer, programming monitor, and test person to have although too detailed for a central repository.

Project monitor systems like SIMON and IMPACT develop a local data base for project management (with secondary objectives of acquiring productivity and reliability data for a particular project.) From these local data bases, reports may be generated -- i.e., the local store is sampled periodically. The amount of data to handle depends upon project size and sampling frequency. Each sample may be quite large unless only recent changes or restricted subsets of the information are reported. The development of a project monitor system represents quite an investment for a project unless a standard system is available and is adopted.

Installation of a distributed data entry system so that data may be delivered to the repository directly rather than via hardcopy or transportable storage media also represents a considerable expenditure both for development and for operation. It does ensure direct, at-the-point-of-occurrence data and immediate acquisition. It is timely enough to serve for management control purposes (but again use might be inhibited by privacy issues.)

The difficulty one service experienced with data acquisition was noted in Section 2.5. One of the reasons that data collection proved intractable may have been a lack of standardization in the data items collected and the form in which it was submitted. Data automation has two potential solutions to offer: (a) the adoption of standard tools and report forms or (b) recognition and extraction of pertinent data from differently structured tools and reports. The first solution is undoubtedly most easily implemented technically and most economically feasible, but it may be difficult to persuade all contractors to adopt the tools and to implement the tools on all pertinent machines. The second approach is more versatile, but is accordingly more difficult and less economical to implement and operate.

Project size may be a significant factor in selecting manual or automated data collection procedures. To provide a full set of automated tools to a small project where the tools have to be fitted to a new computer might cost more than the project itself. However, an input terminal and dial-up transmission service might readily be cost effective for interaction with a regional system monitor system or the central repository.

2.5.3 System Production Goals

Any software production goals or system requirements that call for more than normal software characteristics such as high quality, invulnerability or security, or increased life expectancy will increase both software costs and the costs of ensuring that the goals have been met. Quality assurance procedures will be instituted to monitor the special features more closely and new data types and an decreased granularity both in time and in detail may be called for. In this instance, however, the data acquisition efforts are not likely to meet great resistance and the additional costs will probably be bourne by the application contract. Such special system objectives are likely to increase in importance in the future and may provide increased emphasis for project monitoring capabilities. Such developments could be enhanced by data collection system activities.

One cost factor that might be looked at results from Repository reactions to special types of data resulting from monitoring for special production goals and the cost of changes to standard data collection formats and procedures. Each time some new facet of the software development process is closely inspected, some new data types may result. Procedures for storing these and for retrieving them must be available in the data base management system, but if the system is suitably general, additional costs may be minimal.

2.5.4 Indirect Costs

Increasing the instrumentation of the software development process has secondary costs in delayed schedules, employee irritation and distraction of attention from technical matters. One of the chief advantages of data automation is the minimization of such impacts and the shift of data from the personal to the abstract. Not all data collection can be made objective and automatically or semi-automatically collectable. Filing progress reports and justifications for changes and slippages do intrude on technical activity. Some contracts require monthly, quarterly, and semi-annual reports (all

somewhat different) and may necessitate a fair expenditure of time. This time is usually absorbed by time scheduled for technical activities and even if not, may add a day or more of null activity to ongoing tasks. Employees are irritated by the "extra" work and evince resistance behavior and even turnover and lowered morale in severe cases. Interrupting a technical task in the middle to turn to another, perhaps distasteful task, is distracting and some time may be lost in reestablishing a line of thought, getting reinvolved and recalling what was done before.

Little or no hard data exists to substantiate these assertions. The information is experiencial and subjective and may even be more apparent than real, but so long as the workers believe these allegations to be true, some impact on the work will result. To minimize these costs, data collection needs to be as objective and as automatic as possible and demand little real thought or effort on the part of the worker.

2.6 SYSTEMIC PROBLEMS

There are some problems for a software data collection system inherent in the normal behavior of systems. These include delayed responses, filtering effects, averaging and summation, forecasting efficiency and stability. These effects have been mentioned in relation to prior subjects, but need some consideration as factors in the efficiency of the data collection systems.

2.6.1 Delayed Responses

Time delays arise in every stage of the activities of information-feedback systems - delays in decisions, transmission, processing, reporting, and reacting. These delays can result in inappropriate responses, oscillating (hunting) behavior, and other anomalies. In many manual reporting systems the reporting period (sampling rate) is monthly, so that some events are nearly a month old by the time they are reported (and some information is distorted by being summed or averaged over the month.) Further, it is likely to take a few days to prepare reports and if there are several levels of reporting,

some delay while information is integrated and filtered up and down the reporting hierarchy. Quite often it takes several days for the reports to be digested and a management decision to be made. By that time the data on which the decision is based may be upwards of two months out of date, the situation may have improved greatly or deteriorated out of sight. The reaction will be inappropriate, either over correcting an already improving or causing a bad siutation to be worse.

Improving the speed and automaticity of the reporting and control process corrects to some degree the ill effects of delayed response. Although close and accurate control of the software development process is more a management than a research problem, there is little doubt that inappropriate management control can introduce a large element of error variance into the data that is to be used for research. Such uncontrolled sources of variance can overshadow the real effects of experimental comparisons and cast doubt on, or obscure results. For best results, it would seem desirable to eliminate long delays in the control cycle.

2.6.2 Information Distortion

Information is the input to decisions. Any factors that influence the information flow or change the information can affect the decisions. Information that passes through several management levels is usually modified by averaging procedures, by summarizing many transactions into composite data, by totaling many pluses and minuses, by applying the filters of differing standards of importance, personal biases and prejudices, internal politics, past history and experience and optimistic and pessimistic viewpoints. Too, there are a few "honest" errors and some random noise and external influences in the data and affecting the decision-makers evaluation of the data.

For best research results and for best management decisions, it is highly desirable that those distoring influences be kept to a minimum. It may be impossible to remove all subjectivity, but as much objectivity as possible should be sought. This could include removal of much of the intermediate processing that filters, averages and combines data. Unfortunately, this may result in a flood of detailed data that is difficult to turn into information without the operation of the intermediate process.

As a simple example of what can occur, take the instance of schedule slippage. Say that at a monthly reporting session it is found that many of the activities are a day or two behind schedule. On an average, this may not look too bad - only a day behind time. As a total it may not look too good - one or two man months in the hole. But, assuming task interdependencies such that some crucial tasks later may delay the start of several others, the true impact may represent many days behind and eventually many man months already lost or threatened. Distorted data may easily lead to erroneous decisions both in management control and methodological research.

2.6.3 Instability

Instabilities in the software development process may result from both instabilities in the processing system and the control system. Personnel turnover, requirements changes, and changes in management and key personnel can be very disturbing to the system's efficiency and, of course, to the data collected for comparactive studies. Especially disturbing can be instabilities in the control system. It may not always react in the same way; it may react violently to minor exceptions and not at all to others. It may demand much extra data at one time and ignore submitted data at others. These inconsistencies can be disturbing.

There may be amplification affects or oscillating effects. For instance, a mild reprimand at one level may be blown into a major catastrophe by successive feedback loops. And, with distortion, a comment made at one level or place may be well nigh reversed at another. In designing control systems for some purposes, designers may deliberately include "dampers" to compensate for overreactions, "amplifiers" to compensate for underreaction and "rectifiers" to bring response back in line with the stimulus situation. With people as processors, these compensator actions occur but with little regularity or discipline. Hence, instabilities in the control system and instabilities in the processing system are major sources of costs and of unreliable data.

2.6.4 Forecasting Efficiency

Schedule and cost estimates and estimates of product characteristics are normally made at a very early point in the system development cycle when all aspects of the system are very ill-defined and difficult to assess. There is a great deal of uncertainty in such forecasts that is not always dispelled and, in fact, original estimates frequently get set in concrete and all subsequent performance evaluated against the original estimates. This too leads to imprecise and unreliable data for both management control and research purposes. For best results project planning and forecasting of schedules, costs and characteristics should be iterative; estimates should be adjusted as better, more accurate information becomes available. This does, however, heavily impact the sort of contracting that can be done. Cost plus contracts are readily adapted to such cyclical reestimating, but procurers feel, perhaps with some justification, that contractors take advantage of such contracts to cover their own inefficiencies. That is, there is no incentive to keep costs down nor to operate efficiently. However, Cost Plus Award Fee and Cost Plus Incentive Fee were invented to provide such incentives and, with appropriate project monitoring, should prevent unduly inflated costs and enable more reliable developmental data to be collected.

3. RECOMMENDED DATA COLLECTION PRACTICES

Although the software development community is aware of many of the difficulties that exist for software data collection, recommendations for over coming the difficulties are few and often difficult to implement. Some suggestions, such as the establishment of standard terminology and measures, may be dictated for some contracts but getting industry agreement is not easy and enforcement without cooperation almost impossible. Others, such as the creation of many automatic data collection and programs evaluation tools (a) are state-of-the-art and (b) require a concerted effort to design and develop tools that can be used across a broad spectrum of projects. The benefits of adopting practices that would counter the difficulties need to be evaluated in terms of their feasibility and costs.

3.1 INSTRUMENTATION EFFECTS

Although in a special, controlled experimental setting instrumentation effects may be offset by proper experimental controls, in a general repository there is no way of ensuring that the effects have been compensated for in this way. While the increased motivation and productivity induced by the Hawthorne Effect may be desirable phenomena, the effects interfere with the interpretation of experimental results. It may be impossible to conceal the fact that an experiment is being conducted when new methodology is introduced and data collected, but if the data collected is standard and objective and collected without fanfare, at least the instrumentation does not contribute greatly to the biasing affects. For most comparable results, data for the methodologies being compared should be collected under equal biasing or motivating conditions. In asking questions of the repository, something needs to be known of the conditions under which data were collected in order to make realistic evaluations.

The negative effects of instrumentation - the other half of the Heisenberg Effects - are not likely to be as biasing as the positive effects, but again data collected under different conditions of instrumentation - e.g., manual versus automatic collection - may show larger variation due to instrumentation than to the methodology compared. If the data are collected in such a manner as to arouse resentment, cause significant delay, or interference with mentation, performance may be significantly depressed. Close surveillance may also cause the worker to be more cautious and careful and to work harder. Hence, instrumentation should be as objective as possible, require as little effort as possible on the workman's part, and be as standard as possible in measures and application from project to project. All effects of being part of an experiment will not be avoided since much of it will be independent of data collection, but hopefully the effects due to instrumentation will be minimized.

The principles involved in counteracting (or taking advantage of) instrumentation effects seem clear:

- *Software data collection should be done as mechanically and automatically as possible*
- Preparation of progress reports and briefing by technical personnel should be minimized
- Measures are preferably taken from products and from the processing flow independently of the processors
- Where processing personnel must be involved in the measuring process, the observations should be taken in a way that will enhance the motivating effects and minimize effects leading to employee resistance and disturbance

Objectively observable events such as the delivery of a product or the completion of a task can be detected without disturbance of the workman. The quality and characteristics of the product may be evaluated independently of his efforts, although some cost is involved in the measuring. Problems, errors or substandard work can be recorded and fed back to the worker by written comment. (This is normally done by Configuration Control Board action items or test team discrepancy reports.)

When technical employees are asked to provide data, the collection should be as short and factual as possible. Automated methods of presenting fixed information should be employed with the person providing only the variable data such as hours worked, tasks completed, etc. Where extensive input might be required from technical person, its generation should be posed as a technical or problem solving task rather than a management information task. Progress reports should probably be prepared by management or staff people rather than technical personnel. Emphasis should be placed on deliverable products that can be evaluated rather than on preparing reports of current status.

The project may take advantage of Hawthorne Effects to increase productivity and to increase feelings of the importance of the work and one's contribution to it, but it must be explicitly recognized that there will be no comparability to data collected under less facilitating conditions. If Hawthorne Effects are to be avoided, data collection must be routine and the collection methods applied equally to all projects. It should be noted that it is much easier to collect data whose immediate utility is obvious, and more difficult to obtain data that has no apparent immediate use. Thus it is much easier to collect data of all kinds under experimental conditions while resistance may be expected to providing data that may never be used.

Hence, to provide optimally reliable measures it is recommended that standard project monitors, tool instrumentation and code validation and verification tools be developed and used to collect a standard set of measures. The project monitor must provide capabilities for recording plans and performance accounts for:

Workbreakdown Structure

Software Configuration

Configuration Modification

Discrepancy Correction

Schedule Adjustments

Resource Adjustments

Although averaging, summarizing, filtering and other data manipulations may be required before final storage in a Repository, if possible they should not be done by subjective and biased individuals. The monitor system should provide for standard transformations; otherwise raw data should be forwarded.

3.2 RESISTANCE

The recommended solution to overcome user resistance to management monitoring is to (a) gain developer commitment to project goals (a mission-orientation), (b) engender participation in setting goals and standards, (c) openly present and discuss management views and requirements (d) openly analyze required performances and any changes thereto, and (e) give immediate and public recognition for the contributions of subunits. Conversely, recognized standards must be applied equally to all participants and management must insist upon their being observed - participants must know what is expected of them and management must react immediately if expectations are not met and grant recognition when they are.

While project monitors may follow these procedures and may establish them as standard practices, actual performance may fall short of the desired end. Managerial eclat is required as is sincerity and belief in the goals of the organization. In truth, very few organizations have ever managed to make these practices work consistently. There are always those in the chain of command who are less than completely sincere, who autocratically demand performance rather than soliciting it, who conceal their machinations to gain advantage, who are punitive rather than cooperative, who are too weak to punish or who are chary of awards and recognition. There are some mechanics that may be used to further the attainment of the goals. While these will not succeed in the absence of management commitment and ability, they do support attainment.

3.2.1 Mission Orientation

A project or team organization already has a leg up on mission orientation. There is a singleness of purpose and a recognized goal and, hopefully, a feeling of belonging. The ultimate goal should be stressed, especially in terms of the final utility or benefit. The relationships and contributions of the many intermediate goals and commitments that will have to be met to attain the final objective should be clearly established and stressed. Commitments should be publicly recognized, by company and subunit and by individual if that can be done.

The project mission should be given recognition as obviously as possible, by talks, news releases, and published papers. Such publicity should be known in the organization before release and should have the cooperation and collaboration of team members. That is, the team member should feel personally involved; an impersonal announcement or one in which the individual has little involvement does little to engender mission orientation.

Having symbols with which the employee can identify also helps whether this be a slogan, a piece of hardware, or a poster. This is one of the reasons that programming personnel search so avidly for a good acronym for programs and projects. There is a tendency to denegrate acryonym searching, but since physical symbols are largely lacking for software projects, a good project name to identify with is quite important. Posters are also effective; they can make good wall decorations and serve to hold, symbolically, project goals before the team at all times. Posters for software projects must either have symbolic content or depict equipment from the weapons system or operation associated with the software. Posters are, of course, used for sales purposes; it is as important to sell the project to team members as it is to the public.

3.2.2 Participation

There are many ways of inviting participation of the project members in making decisions concerning project goals and performance standards. Regular coordination meetings, frequent consultation on problems, membership on interface working groups and configuration control boards, and participation in product reviews are among the few. Plenty of short term goals need to be set and commitment toward attaining them solicited from project members. Any specific or special objectives should be played up and emphasized, and the roles to be played by various people recognized and their opinions and commitments sought.

On a team level, using a "walkthrough" of a proposed analysis or design can be a more effective tool than reviewing published documents. The participation of members is obvious, feedback is immediate, and recognition is granted. At a higher level, coordination meetings serve the same purpose. To be effective, the group leader must keep his eye on the main goal - commitments to specific performance objectives. The commitments made should be recorded and verified. It is much easier to get commitment to goals that require special effort if all can see that everyone is engaged, depending upon one another and each is putting out at his peak capacity.

3.2.3 Candor

Frank and open discussions of management views and requirements should be made. Views should not be hidden just because they might possibly be unpopular. The project team should know exactly the performance that is expected of them and the potential awards for meeting objectives or punishments for missing them. There should be no unpleasant surpirses and no unreasonably gratifying ones. All awards should be earned - a cheap victory is seldom valued or appreciated.

In the data collection aspect, the reasons for collecting any data and the uses to which it will be put should be made clear. If the intent is evaluative, this should be evident and admitted. People will be discouraged if the data they supply is not used or meaningless or if it is used in unexpected ways (as to chastize).

In brief, making known the expected performance is half the battle in getting commitments to these goals.

3.2.4 Justifications

Making the reasons for asking for a particular performance apparent and discussing the ins and outs of decisions and changes before making them has the benfit of gaining understanding and concurrence in decisions. Having relatively formal change procedures, for example, has the benefits not only of avoiding some unnecessary and trivial changes but makes sure that there are good reasons for the change and that people understand them.

When the change that is required is increased productivity on the part of some subunit or individual, the need for the improvement should be presented and suggestions sought for meeting the need. If an unassisted commitment cannot meet the problem, cooperative effort to find alternatives should be sought.

3.2.5 Recognition

Recognition really implies recognizing both good and poor performance. To plan awards and punishments in advance may take real effort. It is distasteful to plan for or talk about failure and the rewards therefor. Some people are almost superstitious about it as if admitting the possibility were to deliberately invite it to happen. Unfortunately, this also leaves the contract monitors with no options to choose from if provision hasn't been made to punish poor performance. But equally, the contract monitor needs some special incentives over and beyond "fair pay" to reward good performance.

Recognition needn't cost a lot, however. Public praise and private admonishments, letters of appreciation and approbation, pins and plagues, and published performance standings all cost little more than the effort to issue them but are often appreciated much more than cash incentives.

Part of the impetus for setting many interim goals and convening many coordination meetings is to provide opportunities to grant recognition.

3.3 COST FACTORS

For most projects, the driving force for the collection of software development data is management control. The data collection system must be fitted to these requirements, both in the granularity and the variety of measures. Unless the project is an experimental one for which collecting evidence in support of a technique is a major objective, both project members and management are likely to resist "superfluous" data collection efforts.

The research need for accurate and reliable data implies finer granularity, more meaningful and explanatory information, and greater objectivity of measurement. Development of standard data collection requirements is a step in this direction, but the greatest advance is likely to be found in data automation. Instrumentation of programming support tools (compilers, operating

systems, etc.), development of new tools (design languages, programming support library techniques, code auditors and verifiers) and the use of project monitor systems results in as fine granularity as desired and brings data collection very close to the source of the data. The data so collected is quite objective, accurate and reliable. Automatic collection results in minimum interference with work on a large project may more than save enough time and money to pay for the development of tools and insertion of instrumentation. For small projects, developmental and operating costs may be disproportionate to total project resources. In either case, it would seem expedient for the Repository to develop the data collection tools and provide them and operating funds to all projects. Not only would this tend to overcome user objectives but would establish de facto data standards. All data collection cannot be automated, but it can be regularized.

3.4 SYSTEMIC PROBLEMS

Automation seems to be the best answer to overcoming response delays and to establishing standard handling of the data and overcoming biases and prejudices. To improve forecasting efficiency, a new approach to project planning and plan maintenance is desirable. Contracting must be flexible enough to permit frequent plan modification and project monitoring must be forward-looking more than looking backward at project history.

The hierarchy of reported data and the algorithms for combining, filtering and averaging data should be established. Since we may not have sufficient information at the present time to specify these, some of the later research effort might be directed at evaluating filtering and other processes and at defining amplification damping and rectification effects.

4. OPERATIONS MANAGEMENT

Although the management of system operations, including trade-offs for cost-effectiveness, is an important aspect of systems, it has not received the attention it merits. There are models for making choices among alternatives in Sharpe [51] and Johnson [52], but there is little substantive data to insert into the models. Hence, the analysis of operations management for the data collection and repository system rests heavily on the personal experiences and opinions of operations managers available to the project.

If the design and operation of the data collection and repository system is considered as a whole, including the application of tools and techniques helpful in achieving and improving efficiency of operation, the normal framework of systems theory will apply. This analysis will cover the operating environment, the functions performed, design (organizational) alternatives and operational trade-offs, and will result in recommendations concerning the organization and operation of the data collection system.

4.1 OPERATING ENVIRONMENT

The operating environment of the system reflects the functions performed, the objectives, constraints and performance requirements for the operation, the design and structure of the system, and the magnitude of the system operated. These are each subject to the ultimate decisions concerning the implementation of the system.

Basically, the data collection agency will be charged with collecting data on forms or in machine-readable form, entering the data into the data base, and creating reports from the data. It may also be charged with the maintenance and improvement of the data collection system and other support activities. The exact structure and detailed responsibilities of these functions depend upon decisions made as a result of this data collection study and other studies of the proposed system.

The level of performance demanded by system users impacts the functional operation by stressing their relative importance and the degree of effort directed toward them.

The agency must be able to handle a data base that is being used for both project management and research. The project management use requires timely service and a secure system to protect project management data from illegal access. Research users need a variety of detail on many projects over a period of years but must not be given data that would compromise the privacy of a data contributor. In order to satisfy these users, the agency needs to be flexible in size, able to grow, service oriented, and staffed by careful, conscientious people.

The exact structure of the agency will depend upon decisions made about the operation of the data collection system. These decisions concern:

- data entry configuration
- data storage and retrieval software
- size of the system

The data entry may be performed at the project site or at the central site. Data may be collected at the project site automatically and sent to the central site in machine readable form. It may also be manually entered at the project site using a terminal or key to card/disk/tape device. Completed forms may be sent to the central site for data entry. The forms may be ready for key entry or they may need to be edited or transcribed before they can be entered. In short, the level and importance of the data entry function for the data collection agency are highly dependent upon the mode of data delivery to the agency.

Data entry is further complicated by the storage mode and the sophistication of the data storage and retrieval software. Some manual mode storage of system specifications, engineering change documentation and other original copy must be anticipated. (Project environment and performance data could be stored manually but it is not recommended.) A simple, batch-operated, off-line system places different demands on operators than a complex, on-line, interactive system in terms of the relative skills and training that are required and the sorts of error diagnosis and correction that are practiced. Further, the type of storage hardware, the number of data base storage locations, and the degree of sophistication of the data storage and retrieval system will effect the type of person required in the data collection agency.

The original size, speed of growth and ultimate scope of the operation impact job composition and specialization potentials. At a conference of the associated contractors for the RADC Software Data repository in January 1976, the probably initial operational capability of a pilot facility was defined as a Honeywell H6180 or H6181 computer, a GCOS or MULTICS operating system and an existing data management system (IDS or FSA). Only specific research problems and the resulting limited number of source data suppliers will be involved in the data collection effort. From this limited base, if the pilot operation proves the system a feasible and profitable concept, operations may be expanded greatly to include all Air Force contracts, the provision of project monitor, data collection and other programming support tools to many contracts, and more automated data entry techniques. Hence, a certain growth potential and adaptability must exist for the data collection agency.

4.2 FUNCTIONS OF THE AGENCY

The data collection agency must perform five distinct functions. The exact duties required of each function will depend upon the data entry configuration, the data storage and retrieval system, and the size of the system.

The five functions are:

- management
- data entry
- operation of the computer and the manual storage (library) facilities
- reporting
- system support

4.2.1 Management

Management involves the overall coordination of the operation of the data collection agency. Part of this overall coordination includes responsibility for liaison with higher RADC management, with data suppliers (contractors) and repository users. If the volume of data collected is small and manual data collection and data entry techniques are practiced, this job may be performed by a relatively inexperienced manager. A higher volume of data re-

quires a longer staff, more coordination points, and more system hardware and software, which, in turn, requires a more experienced manager. In view of the importance of persuading Air Force and contractor personnel to use and to cooperate with the agency, a relatively senior and able person is highly desirable.

Once the staff assigned to any one function grows beyond five persons, managers of functional areas will be required. The same criteria of work load and customer interaction requirements may be applied to selecting appropriate management personnel.

If the agency is managed and operated by RADC personnel, the liaison responsibility between the data collection agency and RADC higher management and Air Force project monitoring agencies will be somewhat reduced. If the agency is operated by an outside contractor, the assigned facility manager will have to work closely with RADC to ensure that all mutually agreed upon goals are achieved and contract commitments met. Liasion with data suppliers will be somewhat more difficult with an outside contractor, but SDC experience with the Air Forces' Sattelite Control Facility Computer Program Development Library and the Army's Ballistic Missiles Division Quantitative Data Base indicate that equible relations can be maintained and operations kept adequately efficient.

4.2.2 Data Entry

The data entry function is very dependent upon the data entry configuration. Data entry may be performed at the project site as well as centrally. If forms are sent to a central site which need editing, a data analyst is needed to scan the forms for consistency and to convert free-form entries to standard input formats. If a data analyst is required, he could perform the data entry by using a terminal or transcribing the data to a form readable by an optical character reader. If the forms are subject to little or no editing, key data operations may convert the data to machine readable form without great assistance.

Correction of rejected updates is also handled by the data analysts. When

interactive data entry is used, many errors will be discovered and corrected before the data is accepted but others will not be discovered until the data base is updated. When data is keyed with minimum editing, errors will be discovered when the update is made. As many errors as possible should be corrected by the data analysts; some errors may require that the input forms be returned to the source.

The number of analysts and/or keyboard operators will depend on the data load and point of data entry. If data are entered at the project source, either directly to the control computer or onto a transportable medium, fewer people will be required at the central location. While a relatively small data entry staff may suffice for the pilot facility, the expected expansion will cause this staff to grow disproportionately faster than other functional elements.

4.2.3 Operation

Operations involves the running of jobs for creation, maintenance and utilization of the data base. An auxillary operation is the maintenance of a document library and file facility. This latter is particularly important if program listings and the outputs of program analyzers and automatic collection tools are stored here rather than in machine-readible form. Engineering change (ECP) processing documents and other source documents may also be filed here for reference or for future storage in the machine if the need arises.

It is not expected that the initial level of operation of the pilot facility will be great enough to justify either a dedicated computer or more than a one shift operation. On a shared basis, the operations function for the computer will be performed mainly by a general computer center staff rather than by a specific data collection agency. Job coordination and dispatching is still necessary under a batch mode of operation, but may disappear (into the computer) if an all online mode of operation is adopted. The job controller (or a job monitor in the executive system) will also keep a log of operations and act as an expediter of delayed jobs.

Once the pilot operation period is past, the need for additional shifts to

match the input load (especially if national teleprocessing operation is undertaken) will probably arise. It is not expected that the demand for reports will grow at a similar rate, but it may become substantial. If the load grows to the point that would justify a dedicated computer, the data collection system operations agency should take over the computer operations.

Although it is possible for the library facility to be operated by a general purpose library or files facility, it is equally conceivable to perform as a dedicated facility from the pilot operation on. The chief advantage of the general purpose facility is the availability of full-time professional librarian personnel. Such availability may be limited or not justified by a separate, dedicated facility. In any case, one or more accession and filing clerks will be required.

4.2.4 Report Generation

The report generation function involves producing reports for research and management purposes. Some of the reports generated will be standard reports, some will result from requests for specific analyses, some will result from consultations with research projects, some will be subsets of data for private analyses, and some will be those required to support the operations of the data base. While standard reports take a minimum of technical competence to request once they are established, the initial difinition of these reports and other, non-standard reports will take more skill. Set-ups for analyses and consultations with customers will take somewhat different knowledge.

For the pilot facility, while all these functions will exist, it must be expected that they will operate at a relatively low level. For an enlarged, completely centralized facility, with all requests for data funneled through the analysis group, a fair number of relatively highly qualified persons will be needed as consultants and to set up and prepare the analytic requests, run simulation programs, and perform similar duties. If users have on-line access, however, quite a bit of the analytic function may be displaced. At the same time, user education on the report generation language, the logical structure of the data base, and the information potential will increase. If RADC offers support services such as developing models for reliability analysis

and productivity, some instruction and consultation on the use of these may also be expected.

4.2.5 System Support

System support includes (a) the administration of the data base and (b) the maintenance of the data base management and data collection programs.

The data base administrator is the guardian of the integrity of the data base. He is responsible for the detailed knowledge of the data base structure and composition. Operationally, he is responsible for restarts and reload if a failure or malfunction occurs and for reconstituting the data base if it must be modified. If new data items and data records are to be defined, or if existing stores are to be restructured, the data base administrator will work with analysts and users to define and design the new or revised data structures. He is responsible for creating and entering data definitions, for specifying data entry formats and file storage formats, and maintaining the data base catalog. He is responsible for working with users to define logical (application) views of the data base and declaring these to the data management system and will work with the report generation people to define catalogs and listings for analytic and research use. He will be the principle trouble-shooter in case of failure or malfunction in the data itself. The number of persons required to perform this function depends in part upon the size and complexity of the data base, the level of modification activity and the number of logical views of the data base that exist, but at least one trained person and a backup will be required. The administrators must be thoroughly knowledgeable about the data base management system, data entry and report requirements, and the physical and logical structure of the data base.

Program maintenance is responsible for the maintenance and modification of the data base management programs, analytic and report generation programs, simulation programs for reliability and other models, and any data collection, project monitor and programming tools that the system might supply. It will also be responsible for generating new tools and programs. The size of the staff required and its organization will depend in part on the size and sophistication of the data management system and other support programs, but

also in part upon how the support task is done. If new programs are purchased as packages or written by contract personnel, fewer programmers are required. It is usually desirable also to have program suppliers maintain the processors. However, even though the bulk of generation and maintenance is performed elsewhere, the facility should maintain some support staff for trouble shooting and testing. Further, since the data collection system will be operating in a dynamic, research-oriented environment, there will be a continual flow of new requirements for programs, hardware and system reconfigurations. New items and new releases need to be at least installed and benchmarked before operational use. Program support should perform these duties even if new programming and maintenance work are minimal.

4.3 OPERATIONAL ALTERNATIVES

The choice of an agency to operate the data collection system must be based on criteria of flexibility, control, productivity and cost. The options available for selection are civil service, airmen, civilian contractors or some combination of these possibilities. Each option must be evaluated in terms of the functions to be performed and the criteria to be met.

It is expected that the facility will begin with a pilot facility at RADC with existing equipment and software but may later expand into a more sophisticated and extensive facility with a potentially dedicated computer and operating and data management systems. Distributed and local data bases and processors are remote possibilities; if these options are taken, separate, additional facilities decisions will have to be made. Based on this expectation, these assumptions are:

- Size. The initial facility will be small on the order of a half dozen part time employees but has potential of expansion to a moderate size of 25 to 30 persons plus computer staff.
- Location. Personnel will reside close enough together to foster good communication. If housed separately from RADC or if the computing and support personnel are housed separately, data communication lines will connect the facilities.

- Shifts. Initial operation will be on a single shift basis. If the facility goes to online operation with telecommunication lines, a second and third shift update operation may be instituted.
- Functional Specialization. On initial operation, each person will be expected to perform more than one function or to perform a specific function on a part time basis. As the facility expands, functionally specialized groups will be formed.

Figure 1 shows a tentative organization that might meet these assumptions. The data collection staff is shown separately from the computer center staff because, at least for initial operations, the software development data base is likely to be only a part of the center's operations. The figure also shows a potential for contract programming support, allowing the data collection staff to concentrate on the immediate business of maintaining the data base and dealing with data base users and data suppliers.

4.3.1 Flexibility

The operating agency's ability to adjust to change is vital to the success of the operation. The data collection system, and therefore the agency that operates it, will grow and change over time. The importance of the functions will shift as the volume, types of jobs, and required response time wax and wane. Hence, the organization's malleability is of prime concern as a selective criteria.

On the other hand, some tasks are relatively well-defined and unchanging. For instance, while the keyboard and computer operator tasks may grow, the duties are quite invariant. It is the data analysis, report generation and system support tasks that are most affected.

Agencies operated by Air Force personnel are normally reasonably flexible in organization and operation but are plagued by problems of personnel availability and turnover. Although it is not now as difficult to get personnel trained in data processing as it once was, high level personnel are still at

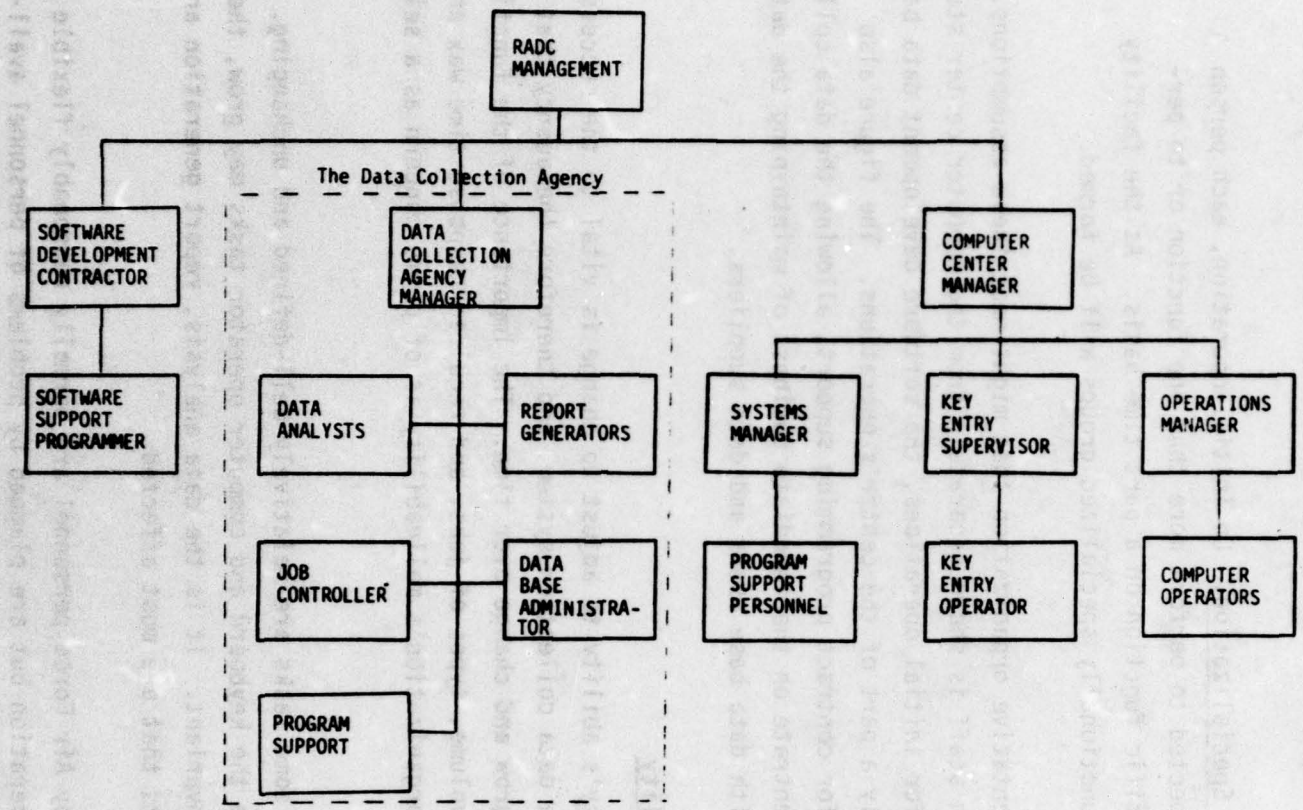


Figure 1. Organization of the Data Collection & Repository.

a premium. Both acquisition of substandard personnel and high turnover (a term of service is ordinarily 2-4 years) will place a considerable training burden on the existing staff. However, computer operators, keyboard operators and data clerks are jobs readily filled by service personnel.

Civil service organizations have better access to trained personnel, but their flexibility is often impaired by restrictions and red tape in adjusting tables of organization and laying off persons with tenure, and by the often bureaucratic procedures necessary to justify a change and obtain approval for them. Stability on the job is quite good for Civil Service personnel and the retraining load less than with Service personnel. Civil Service is judged least flexible largely because any given structure becomes established as the minimal, standard staffing level for requirements that no longer pertain.

Civilian contract personnel are normally quite flexible in meeting changes in contract requirements. Contracts are reviewed and revised yearly and contractors usually have pools of people upon which they can draw for exceptional jobs and to which people can return when specific jobs are eliminated or changed or when special jobs are complete. It has already been suggested that outside programming support be obtained for generating and maintaining software tools. Other things being equal, civilian contract personnel provide the best choice in terms of flexibility.

The option of using a mixed group cannot be ruled out. In fact, using Civil Service or Air Force personnel for relatively stable positions, augmented by contract personnel for positions subject to rapid change might provide the best flexibility for the operation.

4.3.2 Controlability

The responsiveness of the agency to direction will also be of prime importance. It will be the RADC staff that sets the goals and establishes the tasks for the operation and these goals and tasks will change as the data collection system evolves. Further, evaluations of the efficiency and effectiveness of the operation are based on how well goals are met. Control of the operation requires clearly defined tasks and goals that can be monitored and corrective

directions issued if performance deviates from that planned and desired.

Being under the direct command of the RADC organization, Air Force agencies may be considered generally most responsive to direction. However, penalties for poor performance (and rewards for good) are difficult to levy against in service personnel and it may not be politically expedient to be very critical of the operation. Hence, while it is easy to direct, it is difficult to exert leverage if performance is poor.

Civil Service may also be expected to operate under the direct command of RADC personnel. However, while it would seem that responsiveness to direction and exercise of rewards and punishment should be relatively easy, experience would seem to indicate that these operate even less well for Civil Service than for in-service personnel. It is not easy or expedient to remove either Civil Service or in-service personnel for poor performance or for failure to follow orders. Criticism of the Civil Service operation is almost as difficult as for in-service organization.

Civilian contract agencies then are the most effectively controlled, provided that the proper provisions for reward and punishment and procedures for issuing and responding to directive are built into the contract. In short, the civilian contractor is readily controlled by insisting on adherance to his contract and by proper administration of the incentive program. It is relatively easy, both politically and in a business way, to criticize individuals who are not part of the same organization and to get intransigent and poor performers removed.

Administration of a mixed group is probably the most difficult since it entails more diverse and complex political and organizational conditions. Very good success has been reported by mixed Air Force/Civilian teams where the two sets of people are able to work closely together on a project basis. On a long term basis, however, more formal relationships tend to arise and control, except insofar as self discipline is exercised, may be expected to diminish and become difficult.

4.3.3 Productivity

Productivity has several facets. One is the efficiency of internal operations. Another is the quality of the work performed and a third is the effectiveness of the agency in dealing with the projects and agencies involved in supplying data and using the data base. Productivity is partially a function of the level of skill and training of the personnel and experience on the job. It is also partially a function of the responsiveness to control as just discussed.

The productivity of Air Force personnel is inhibited by a variety of factors including the difficulty of finding properly trained and skilled personnel within the service. Frequent turnover is another inhibiting factor and so is the performance of other in-service responsibilities. Lack of responsiveness to the exercise of incentives may also inhibit productivity.

Criticism of Civil Service prodictivity by civilian agencies is almost slanderous and frequently erroneous. Civil Service job standards ensure that adequately trained personnel occupy the positions. Stability on the job of Civil Service personnel tends to be high, but this also tends to restrict flexibility. Liberal vacation and sick leave policies may also cut into productive time. Tenure and other factors may also make Civil Servants unresponsive to control incentives*.

Staff efficiency for civilian contract personnel is apt to be high for several reasons. Individuals are more accountable and more subject to removal than Air Force or Civil Service personnel. The level of experience and training tends to be reasonably high although not so rigidly stated as for Civil Service. Stability on the job tends to be adequate compared to the regular turnover of Service personnel, but less, in general, than for Civil Service.

*An informal study of pooled computer facility operations found keypunch operator productivity approximately 60% of the "industry norms" with a high incidence of error. Blamed were "civil service bureaucracy" and loss of control through centralizations. Some reservations should be exercised in accepting these findings in view of the subjectiveness of the investigation.

Data supplier reluctance to release data is a factor that may operate more strongly for the civilian contractor option than for the other options. Projects are understandably leery of providing data concerning their operations to potential competitors. The data collection agency must use considerable delicacy in handling customer relations in any case, and evidence of mishandling sensitive data, inefficiency, creation of errors, or use of the data for agency advantage is likely to increase user resistance and reduce the productivity and effectiveness of the management agency.

It is, of course, possible for a civilian contractor to overcome this reluctance by the exercise of integrity and the separation of the facilities management operation from all software development contract activity. SDC, for instance, has managed to perform effectively as repository manager for both the Army Ballistic Missiles Division in Huntsville and the Air Force Satelite Control Facility in Santa Monica. Careful attention must be given to keeping the data collection agency organizationally separate from any software development agencies and in exercising objectivity and responsibility in handling sensitive data not only in reference to associate contractors but to disclosure of Air Force data of a sensitive nature. SDC has been able to build the necessary trust in the contractors by being service oriented, careful about protecting data from unauthorized access, and objective about evaluations. Certainly the problem of building trust would have been less for a Service or Civil Service Agency, but from experience the efficiency and productivity of these repositories has been quite good. The agencies have been most responsive to Service needs so that it is felt that the benefits of an outside contractor outweights the problems.

4.3.4 Cost

Cost is a critical factor in the evaluation of potential operations managers. It must be considered in conjunction with the relative benefits of flexibility, control and productivity that the options offer. Cost is not a simple variable but is composed of salaries, fringe benefits, and administrative cost associated with the management of the selected candidate. There are also costs associated with turnover - the cost of acquisition and training and "lost revenue" until the new employees reach a satisfactory level of

effectivity. Office and work space also represent significant costs, but since the operation is assumed to be internal to RADC environs, this is a standard cost for all options.

Of these factors, there are firm data only on direct salary and fringe benefits and these are quite variable. That is, salaries and fringe benefits vary widely with the levels of skill, experience and training, working location and conditions, and other factors. Some cost factors may be assumed to be equal regardless of the options chosen such as office space (on the assumption that the facility will be located at RADC). Some cost factors are ill-defined, such as the cost of turnover. This includes hiring and firing costs, training costs, costs of maintaining some level of personnel replacement inventory, the cost of obtaining clearances (negligible to the agency, but considerable to the government), and discription to the work. Since a departing person takes some measure of information and special knowledge away with him, there is also a "capital loss" in replacing it and him. There are also costs associated with overwork and underwork - overtime and idle time. For Air Force and most professionals this is not a serious concern, but for non-exempt persons it could represent a fair fraction of the basic costs.

Table 4 presents some rough estimates of average weekly salaries and fringe benefits for civilian, inservice and civil service personnel. Data for Air Force personnel are taken from the Airman Pay Guide [53] and a U. S. News and World Report article [54] and the Bureau of National Affairs Policy and Pay Practices [56]. Data for the civil sector was taken from the most recent salary survey by Datamation [57] and the Pay Practices. Data in the table are rough; actual salaries range from 40% less to 50% more depending upon experience (time in grade) and skill level. Fringe benefits also vary, but not as widely. Based in the above references, fringe benefits are estimated at approximately 32% of base pay for private contractors, 34% for civil service and 60% for inservice agencies. At these rates, inservice rates are still somewhat less than similar exployees elsewhere, but not grossly so. (In-service costs were estimated somewhat higher in reference 56 for lower ranks, but this would seem a fair estimate.) This cost data is not definitive enough to have a strong influence on the selection of the type of personnel

Table 4. PERSONNEL COST COMPARISONS*

Personnel Classification	OPTION: Civilian Wky. Sal.	Civilian Frn Ben.	Civilian Tot. Cst.	Civil Service Wky. Sal.	Civil Service Frn. Ben.	Civil Service Tot. Cst.	Air Force Wky. Sal.	Air Force Frn. Ben.	Air Force Tot. Cst.
Data Entry									
Entry	124	40	164	121	41	162	116	70	186
Lead	177	57	234	153	52	205	137	82	219
Tab. Oper.									
Entry	135	43	178	120	44	174	104	62	166
Lead	208	67	274	195	66	261	129	77	206
Comp. Oper.									
Entry	149	48	197	150	51	201	113	68	181
Lead	262	84	346	244	83	237	173	104	277
Programmer									
Entry	211	68	279	235	80	315	184	110	294
Lead	353	113	466	407	138	545	263	158	421
Analysts									
Entry	253	81	334	259	88	347	168	101	269
Lead	402	129	531	440	150	590	313	188	501

*These salary figures are rough average estimates and do not reflect well such crucial factors as experience and skill levels that cause actual salaries to vary by ± 40% about those averages.

best suited to operate the agency.

4.4 CONCLUSIONS

Four criteria were set forth for the evaluation of various operations management agencies: Flexibility, Controlability, Productivity and Cost. The results of the evaluations for the four options of civilian contractor, civil service, in-service and mixed management are shown in Table 5. Although the decision matrix indicates a clear advantage for management of operations by a civilian contractor, the actual differences are small. It is felt that there is a definite advantage for the civilian agency in terms of flexibility and controlability, but the productivity evaluation is based on hearsay. Certainly SDC has had excellent results in working in mixed teams with Air Force, Army and Navy personnel, but both good and bad results were reported in the software development problem survey. (See TM-5542/005, Volume V of this report.)

Civil Service was down graded on flexibility because of known difficulties in establishing new positions once an organization is formed and the practice of freezing allocations as a cost control or budgetary practice. However, Air Force and mixed groups are very close to the civilian contract agency in terms of adjusting to new demands and directions. It is felt, however, that private corporations have a greater supply of experienced, trained persons to draw upon in mobilizating, to meet new situations quickly and efficiently.

The Civilian contract agency was given a plus on control ability only on the assumption that appropriate penalties and awards are built into the contract to provide adequate incentives to follow direction and make it worth their while to meet objectives. Objectives, too, must be very clearly stated to make interpretation of performance unambigious. If directives cannot be enforced or if performance evaluations are open to challenge, the civilian agency loses its advantage and the in-service agency has a clear plus. Civil Service is downgraded because of the lack of an enforceable contract and the general difficulty of dismissal of Civil Service personnel.

There is no doubt that given suitable circumstance the productivity of any

organization can be high. In-service personnel has been given a downgrade due to the interference of other duties, frequent turnover, and current lack of adequately trained personnel. The civilian contract agency has been upgraded chiefly due to the greater incentive for performance - the chance of loss of contract if performance is not good. Some advantage may also exist in the levels of skill and training available to the civilian contractor.

Cost presents an ambiguous situation. Direct salaries for in-service personnel are low and civil service and civilian salaries are roughly on a par. However, Civil Service fringe benefits are a trifle better than for the private sector and in-service benefits are very substantially better. If only direct salaries are considered, then, in-service personnel are substantially less costly. If fringe benefits are considered, in-service are more expensive. Civilian and Civil Service personnel costs are roughly equal.

Since costs are reasonably close for any option chosen, the option providing the most benefits otherwise should be selected. In our evaluation, the civilian contract agency appears the better choice, but this is only true if certain provisions can be met:

(a) Contractor profits and overhead rates are not excessive
(b) The contract provides convenient and efficient procedures for tasking or changing direction
(c) Adequate incentive procedures for rewarding good performance and penalizing poor are included in the contract
(d) Guarantees for maintenance of training and skill levels are provided.

If these provisions can be met, it is felt that a civilian contractor will provide the best service. In our estimation, however, the actual difference is a fractional one; no orders of magnitude exist to make an easy choice. Enough variations exist in salary ranges to obscure all cost/benefit advantages - i.e., a worst case salary picture could add 50% to the salaries quoted in Table 4. A 'best case' could reduce them 30% or more. However, a civilian contractor will be more interested in keeping costs down and remaining competative than the other choices and remains the better choice.

TABLE 5. DECISION MATRIX FOR THE SELECTION OF AN OPERATIONS MANAGEMENT AGENCY

OPTION	Flexibility	Controlability	Productivity	Cost	Rank
Civilian Contractor	+	+	+	0	1
Civil Service	-	-	0	0	4
Air Force	0	0	-	+	3
Mixed Group	0	0	0	0	2

(The reverse of this page is blank)

5. CURRENT MILITARY DATA COLLECTION PRACTICES

If a viable repository of software development data is to be established for Air Force systems, the data collected must not only be reasonably standard, and in standard formats, but be gathered in a relatively standard fashion. Otherwise, the task of filtering and transforming the data to be sure that comparable information in the formats and configurations required by the data base are collected may well be as ungovernable as past attempts to establish repositories have reputedly been. With this premise in mind, current military software development data collection practices may be examined to see whether or not the desirable data are collected and in what form, and where major deficiencies might lie.

The military, and especially the Air Force, began extensive efforts to establish standard configuration management and program control practices in the early 1960's. Over the years this data collection system has evolved into an extensive aggregation of regulations, manuals, pamphlets and instructions that regulate procurement practices. To the degree that these standard practices are actually followed, they may be taken as a general description of the current military data collection system.

The basic requirements for project and configuration control are epitomized by AFSCP/AFLCP 173-5, Cost/Schedule Control System Joint Implementation Guide (C/SCS)[58] and AFR 800-14, The Management of Computer Resources [59]. C/SCS ties into MIL-STD 881 (work breakdown structures [22], and computer resources into MIL-STD-483 [19]. Since these volumes are general, individual program offices have the option to develop in-depth guidance, especially for configuration management. Contractors are required to file management plans, or Computer Program Development Plans, specifying how they intend to manage the project, frequently selecting data items for reports from TD-3, DOD Index of Specifications and Standards. Mitre Corporation is currently preparing an interesting series of handbooks [67] on software acquisition management for Air Force program directors that will eventually cover the gamut of software development at the project monitor level*.

*ESD recently issued an RFP seeking a contractor to complete this series.

Although internal data collection systems may be automated or semi-automatic, reporting to project monitoring agencies is invariably hard copy. Computer produced reports are acceptable and machine readable reports could be produced. Although the most commonly used reporting frequency is monthly, configuration management reporting may be limited to quarterly or end-of-phase reports.

Although deficiency detection is recognized, discrepancy correction control procedures are not as explicitly specified as are those for modification. Since the handling of problems and program errors is a crucial part of any investigation of program reliability, this is one area requiring better definition in the specified standard practices. Where configuration control is being exercised over an existing system. discrepancy control procedures are usually much better established than for developmental projects. For instance, the AFSCF TOR-269(4110-01)-38[60] specifies detailed procedures for handling modifications to the system, including specified organizational elements and a range of control forms.

The basic question is: How well does the existing requirements satisfy software development data collection requirements for the various categories of the following data:

- Project Environment Characteristics
- Project Performance Characteristics
- Product Configuration Characteristics

5.1 ENVIRONMENTAL CHARACTERISTICS

No explicit data items describing the project are collected although much of the information may be extracted from the project contract or proposal. The contract type is known from the contract. The Technical Approach and Statement of Work permit inferences concerning the size and complexity of the project and the methodology used. In the Management Plan, the mapping of the Work Breakdown Structure onto the project organization indicates how the project is organized. If not, the Technical Approach or Management Plan may do so. However, such plans do not necessarily mean that the proposed organiza-

ion is used and the data should be checked by personal observation or specific data gathering efforts.

The type of computer, the operating system, and the programming language are easily determined from reports but other tools are less readily apparent. The mode of interaction with the computer, the sorts of terminals employed, and the physical attributes of the production facility may, or may not, be apparent.

If detailed manpower utilization reports are obtained, it is possible to determine the skills distribution, relative experience, levels of training and familiarity with the application and/or customer.

Ratings of customer rapport are not normally collected, either as interaction modes and procedures or as degrees of cordiality. Even the number of coordination agencies is not always available. The relative locations of client and developer may be inferred, and the granularity of reporting may be apparent to a degree from the CDRL. However, this is all information that must be extracted by examination of the contract, the contract history, and/or by observation.

Evaluations of the quality and granularity (closeness of control) of project management are not normally collected but some assessment may be made from the project mangement plan and the configuration control and project control office proceedings, if available.

Ratings of stress factors, such as adequacy of time, tools, personnel, working conditions, experience, and computer time, power and storage, are not collected. Stress is frequently cited as affecting project performance, but such measures have seldom been taken. The SDC studies attempted to assess some stress factors, but the subjectivity of the measures and after the fact estimates cast some doubt on their validity. (Psychological studies indicate that stress over the short run does not necessarily lead to degraded performance, but does require unusual expenditures of energy to maintain performance. Over the long run, stress does lead to degraded performance and

personnel difficulties.) Currently, it would take a penetrating audit by experienced personnel to accurately evaluate the degree of stress that exists in a project. More objective measures could be developed through the respository by comparing project characteristics with job requirements but such measures are not now collected.

Project stability ratings, such as turnover in management, key personnel, and project personnel, reorganizations, reassignments, amount of system modification activity or stability of requirements, and turnover in project monitor personnel, are seldom collected. Some military contracts do require acquisition and separation statistics on manpower utilization reports. Developers tend to resist providing this information and it is not normally collected.

5.2 PROJECT PERFORMANCE CHARACTERISTICS

Military contracts normally require cost and schedule reports displayed in accord with the Work Breakdown Structure. The general schema for a WBS is shown in Figure 2 (from AFSCP 173-5). The top level breakdown is by major elements (i.e., Configuration Items), and the bottom level is either by tasks, configuration items, or both. Accounts may be further broken down into work packages and/or subtasks. Figure 3 shows sample report forms for cost performance reporting; first, by WBS and, second, by organizational or functional category. Variations of these forms are used by most projects for reporting. A report frequently used on projects using PERT is the Management Summary. (IMPACT's version of this report is shown in Figure 4). These reports show scheduled and actual progress and expenditures and the variances between them.

Resource utilization over time is also reported. Figure 5 is a sample manpower loading report with manpower broken into organizational or functional categories and projected over time. The information shown is planned and actual resource expenditures and turnover of personnel (acquisitions and separations). The utilization of resources other than manpower may be reported in a similar fashion, but is usually kept internal to the project. For research purposes, non-dollar resource utilization reports are often more informative than dollar cost reports.

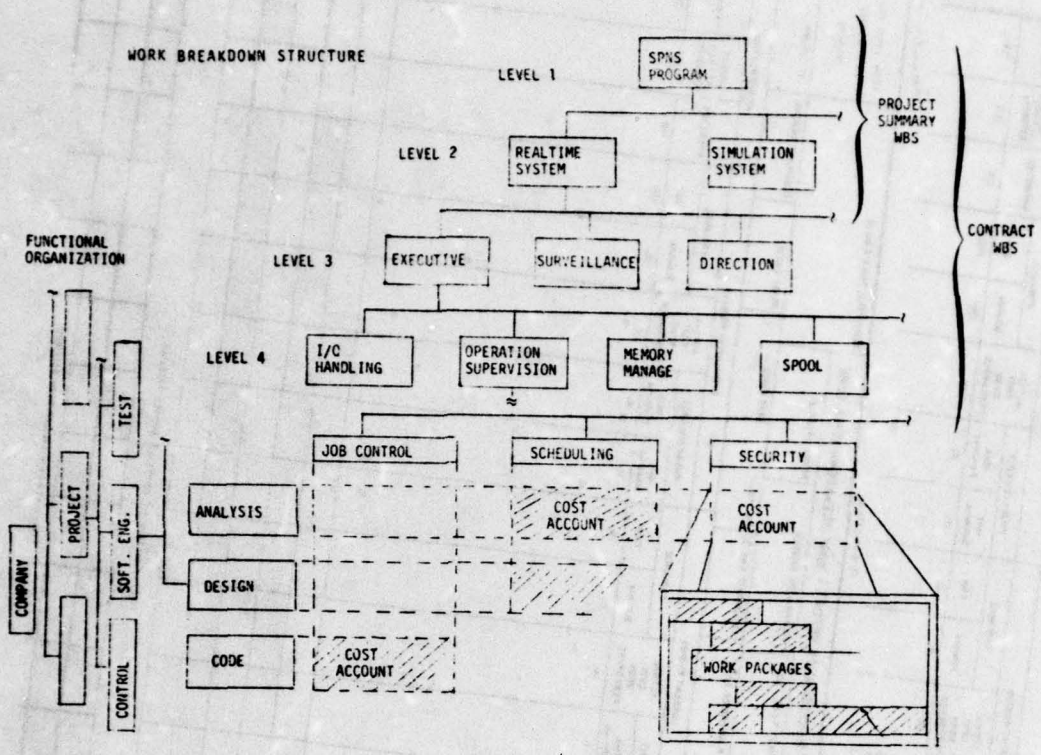
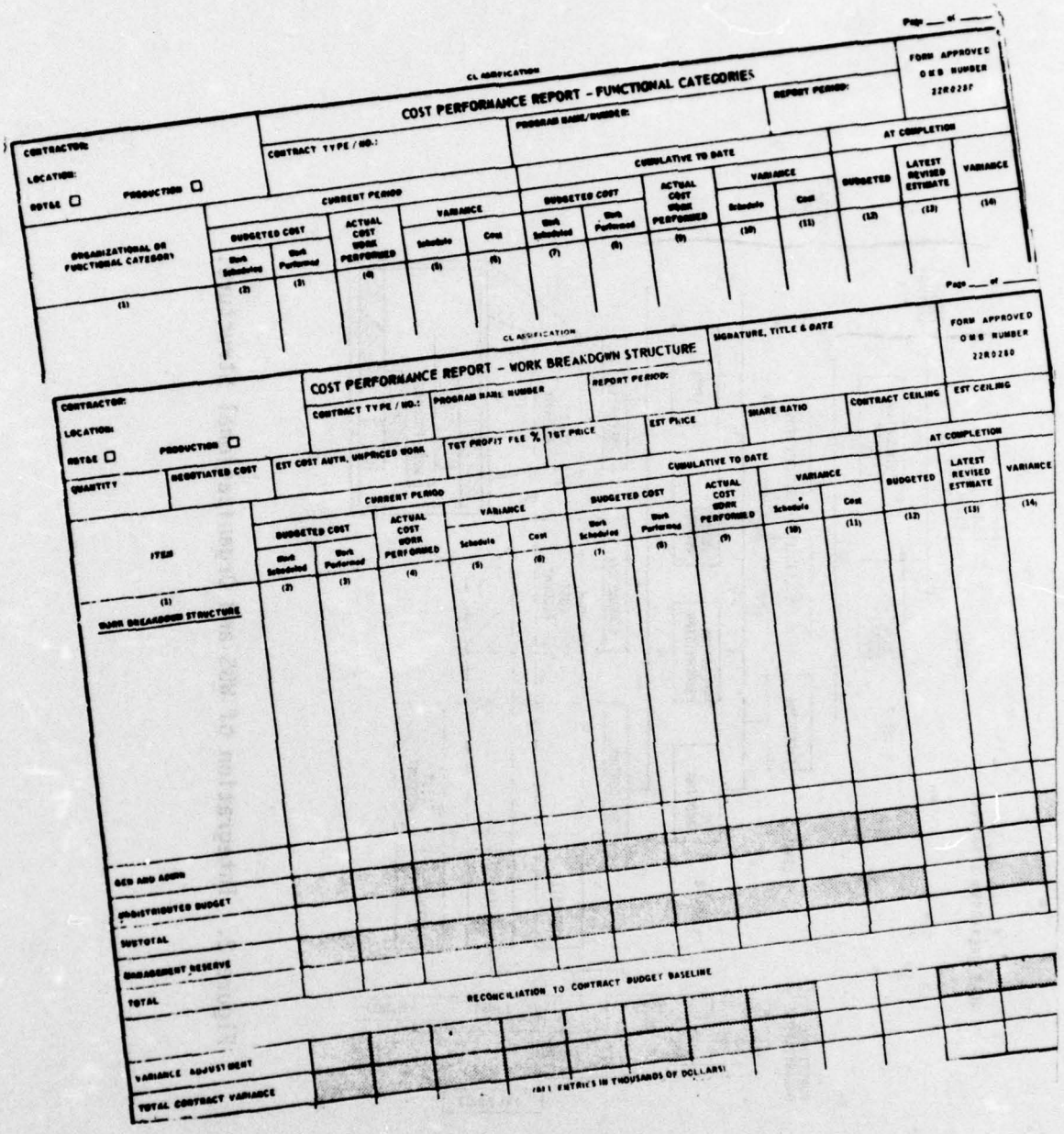


Figure 2. Integration of WBS and Organizational Structure.



Page ___ of ___

CLASSIFICATION

COST PERFORMANCE REPORT - FUNCTIONAL CATEGORIES

CONTRACTOR:
LOCATION:
DIVISION ☐ PRODUCTION ☐
CONTRACT TYPE / NO.:
PROGRAM NAME/NUMBER:
REPORT PERIOD:
FORM APPROVED OMB NUMBER 22R0220

ORGANIZATIONAL OR FUNCTIONAL CATEGORY	CURRENT PERIOD					CUMULATIVE TO DATE					AT COMPLETION		
	BUDGETED COST		ACTUAL COST WORK PERFORMED	VARIANCE		BUDGETED COST		ACTUAL COST WORK PERFORMED	VARIANCE		BUDGETED	LATEST REVISED ESTIMATE	VARIANCE
	Work Scheduled	Work Performed		Schedule	Cost	Work Scheduled	Work Performed		Schedule	Cost			
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)

Page ___ of ___

CLASSIFICATION

COST PERFORMANCE REPORT - WORK BREAKDOWN STRUCTURE

SIGNATURE, TITLE & DATE

FORM APPROVED OMB NUMBER 22R0280

CONTRACTOR:
LOCATION:
DIVISION ☐ PRODUCTION ☐
CONTRACT TYPE / NO.:
PROGRAM NAME/NUMBER:
REPORT PERIOD:
SHARE RATIO
CONTRACT CEILING
EST CEILING
QUANTITY
NEGOTIATED COST
EST COST AUTH. UNPRICED WORK
TGT PROFIT FEE %
TGT PRICE
EST PRICE

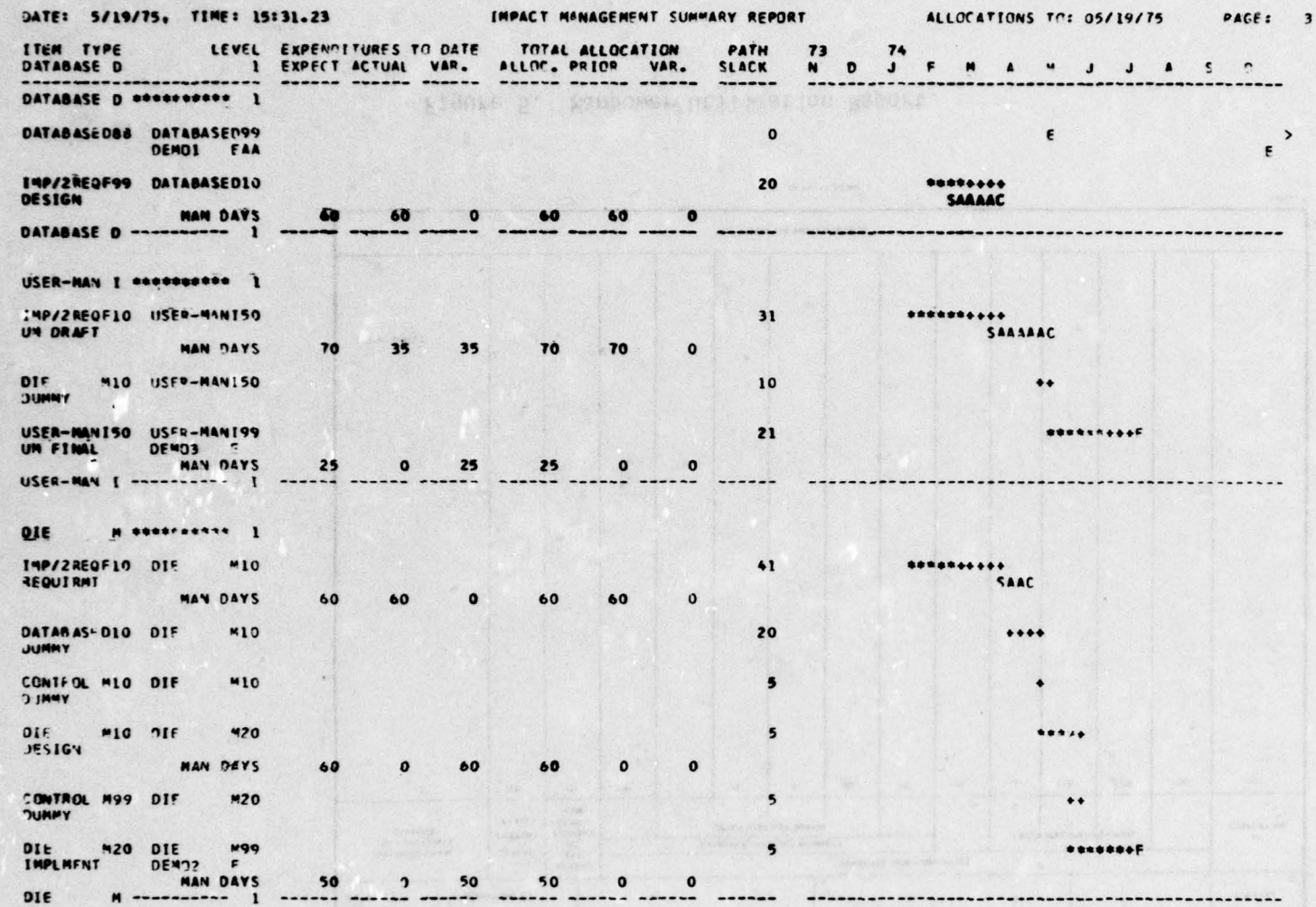
ITEM	CURRENT PERIOD					CUMULATIVE TO DATE					AT COMPLETION		
	BUDGETED COST		ACTUAL COST WORK PERFORMED	VARIANCE		BUDGETED COST		ACTUAL COST WORK PERFORMED	VARIANCE		BUDGETED	LATEST REVISED ESTIMATE	VARIANCE
	Work Scheduled	Work Performed		Schedule	Cost	Work Scheduled	Work Performed		Schedule	Cost			
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
WORK BREAKDOWN STRUCTURE													
G&A AND UBOH													
UNDISTRIBUTED BUDGET													
SUBTOTAL													
MANAGEMENT RESERVE													
TOTAL													

RECONCILIATION TO CONTRACT BUDGET BASELINE

VARIANCE ADJUSTMENT													
TOTAL CONTRACT VARIANCE													

ALL ENTRIES IN THOUSANDS OF DOLLARS

Figure 3. Schedule and Cost Performance Reports.



DATE: 5/19/75, TIME: 15:31.23 IMPACT MANAGEMENT SUMMARY REPORT ALLOCATIONS TO: 05/19/75 PAGE: 3

ITEM TYPE / DATABASE D		LEVEL	EXPENDITURES TO DATE EXPECT	ACTUAL	VAR.	TOTAL ALLOCATION ALLOC.	PRIOR	VAR.	PATH SLACK
DATABASE D **********		1							
DATABASED88	DATABASED99 DEMO1 FAA								0
IMP/2REQF99 DESIGN	DATABASED10								20
	MAN DAYS		60	60	0	60	60	0	
DATABASE D ----------		1							
USER-MAN I **********		1							
IMP/2REQF10 UM DRAFT	USER-MANI50								31
	MAN DAYS		70	35	35	70	70	0	
DIE M10 DUMMY	USER-MANI50								10
USER-MANI50 UM FINAL	USER-MANI99 DEMO3 F								21
	MAN DAYS		25	0	25	25	0	0	
USER-MAN I ----------		1							
DIE M **********		1							
IMP/2REQF10 REQUIRMT	DIE M10								41
	MAN DAYS		60	60	0	60	60	0	
DATABASED10 DUMMY	DIE M10								20
CONTROL M10 DUMMY	DIE M10								5
DIE M10 DESIGN	DIE M20								5
	MAN DAYS		60	0	60	60	0	0	
CONTROL M99 DUMMY	DIE M20								5
DIE M20 IMPLMNT	DIE M99 DEMO2 F								5
	MAN DAYS		50	0	50	50	0	0	
DIE M ----------		1							

Figure 4. Management Summary Report.

CLASSIFICATION

Page ___ of ___

CONTRACTOR: LOCATION: RDT&E ☐ PRODUCTION ☐	COST PERFORMANCE REPORT - MANPOWER LOADING			FORM APPROVED OMB NUMBER 22R3283
	CONTRACT TYPE AND NO:	PROGRAM BASE, NUMBER:	REPORT PERIOD:	

ORGANIZATIONAL OR FUNCTIONAL CATEGORY	ACTUAL CURRENT PERIOD	ACTUAL END OF CURRENT PERIOD (CUM)	FORECAST (NON-CUMULATIVE)											AT COMPLETION
			SIX MONTH FORECAST BY MONTH (ENTER NAMES OF MONTHS)							(ENTER SPECIFIED PERIODS)				
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)
TOTAL DIRECT														

(ALL FIGURES IN WHOLE NUMBERS)

FORM A

CLASSIFICATION

Figure 5. Manpower Utilization Report.

Except for the completion of tasks and the delivery of scheduled items, production statistics are not now generally reported. Sometimes 'percent complete' statistics are included in progress reports, but in the past have proven to be highly fallacious measures of productivity. Some projects divide work into small blocks and modules to increase the fineness of reporting of productivity with fair success. However, difficulty, complexity and familiarity factors often interfere with the comparability of production units. In brief, current reporting practices emphasize cost accounting, not productivity.

Practices concerning accounting for the impacts of modifications on schedules and costs appear quite inconsistent. Figure 6 shows a form used to report cost impacts of baseline changes, but other desirable information is not here, such as schedule impacts, the amount of work discarded, or the number of production units added. However, this information can frequently be extracted from the ECP and Change Report forms.

Except for Error, or Discrepancy or Software Problem Report forms, error statistics, cost and schedule impacts, and seriousness of the problem are not reported. Since the contractor normally must bear the cost of corrections and problem resolutions (or at least hide the costs in other charges) project offices do not require schedule and cost accounts on error correction. However, in a recent IBM Technical Report [15] the maintenance costs of OS/VS was estimated to be over half of the total cost of the successive releases of the system. Some of these costs may be recaptured from problem reports, but it would appear that no conscious effort is now expended to do so.

5.3 CONFIGURATION CHARACTERISTICS

Configuration accounting is directed at WBS or configuration item elements. Since the detailed functional and structural characteristics of the system are recorded in the system specifications, actual control and accounting are exercised over these documents. After delivery, control may also be exercised over the physical programs to prevent unauthorized change. Configuration identification is reported via a Configuration Index, configuration status via a Development Record, and change status via a Change Status Report. Together these reports form a Configuration Status Report. Sample contents

CLASSIFICATION

Page ___ of ___

CONTRACTOR LOCATION INITIAL ☐ PRODUCTION ☐	COST PERFORMANCE REPORT - BASELINE CONTRACT TYPE / NO.:		PROGRAM NAME/NUMBER		REPORT PERIOD:	FORM APPROVED O M B NUMBER 22R0280
(1) ORIGINAL CONTRACT TARGET COST	(2) NEGOTIATED CONTRACT CHANGES	(3) CURRENT TARGET COST (1) + (2)	(4) ESTIMATED COST OF AUTHORIZED, UNPRICED WORK	(5) CONTRACT BUDGET BASELINE (3) + (4)	(6) TOTAL ALLOCATED BUDGET	(7) DIFFERENCE (5) – (6) (SEE PAGE 5)

ITEM	BCWS CUM TO DATE	BUDGETED COST FOR WORK SCHEDULED (NON-CUMULATIVE)											TOTAL BUDGET
		SIX MONTH FORECAST							(ENTER SPECIFIED PERIODS)				
		+ 1	+ 2	+ 3	+ 4	+ 5	+ 6						
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
PM BASELINE (BEGINNING OF PERIOD)													
(LIST BASELINE CHANGES AUTHORIZED DURING REPORT PERIOD)													
GEN AND ADMIN													
UNDISTRIBUTED BUDGET													
PM BASELINE (END OF PERIOD)													
MANAGEMENT RESERVE													
TOTAL													

CLASSIFICATION

Format 3

Figure 6. Accounting for Cost Impacts of Change.

of the reports are shown in Figures 7 and 8. The Configuration Index shows the base documents; the Specification Change Notices (SCN) that transmit modifications to the documents; and the Engineering Change Proposals (ECP) and Change Reports (CR) whose proposed changes are incorporated in the specification of other documents. The Development Report covers the state of progress of the Configuration Item vis a vis the development milestone. The figure is a reproduction of the Record as kept in a paper file; many projects report milestone status in a much more succinct way.

The delivered system is accompanied by a Version Description Document (VDD) that recapitulates the status of the system as delivered. The contents of a VDD are summarized in Figure 8. After delivery, status may be kept on the computer programs as well as on system documentation. Change Notices (CN) or Modification Transmittal Memoranda (MTM) are used to transmit change packages or changed modules. Software Problem Reports (SPR) and Discrepancy Report Forms (DRF) may be added to the list of items that modify the baselined definition of the configuration items, and that are included on the Change Status Report.

In addition to the information that is presented in the Configuration Status Report, much additional information for research may be extracted from the ECP, CR's, SPR's, and DRF's that are used to invoke and control changes to the system. Samples contents of these forms are shown in Figures 7 to 12. Change and error classifications may be derived, and estimates made of the size and difficulty of the modifications. In some configuration control systems, some of the descriptive text accompanying these forms is stored in the data base for listing as part of the Change Status Report.

There are also a number of items that may result from Configuration Control Board activities. These include Action Items, CCB Directives, critiques of specifications, and CCB minutes. Minutes and memoranda usually serve instead of forms to convey this information. Status updates and similar forms may be used to modify ECP's and DRF's. Preserving these update forms keeps a history of development for each item. For instance, a programmed change may be found not to correct the intended error. The update form withdraws the CN

SECTION I. PART I CPCI SPECIFICATION CG7000A - Applications Program (6 Volumes + 4 Appendices)

PART 1. BASIC DOCUMENTATION

ISSUE	DOCUMENT No.	TITLE	ISSUE DATE
Basic	CG7000A, Vol. 1	General	3/2/73
SCN 2-1	ECP 2 ECP 3 CR 3	Change Site Limits ADD DSP Inputs Clarify NCS Reference	5/4/73
SCN 10-1	ECP 12 ECP 13 CR 11 CR 13	Add SIM-2 Function Combine Commit Actions Clarify ADPO Routines Correct Operator Acronyms	1/6/74
⋮	⋮	⋮	⋮
Basic	CG7000A, Vol. 2	Manual Inputs	3/2/73
SCN 2-2	ECP 2 CR 2 CR 4	Change Site Limits Information Transfer Corrections Correct Conversion Equations	5/4/73
SCN 9-2	ECP 10 ECP 11 CR 8	Standardize Identification Acronyms Change Deferred Combat Status Clarify SD Console Restrictions	12/2/73
⋮	⋮	⋮	⋮
Basic	CG7000A, App. IV	Message Formats	3/2/73
SCN 2-IV	ECP 2 CR 2	Change Site Limits Information Transfer Corrections	5/4/73
SCN 29-IV	ECP 23 CR 22	Data Link Message Formats Clarify Lateral/Command Speed	8/7/75

PART 2. APPROVED CHANGES

ECP No.	TITLE (6 Volumes Affected)	APPROVED
38	Automated Weather Inputs - Vols. 1, 2, 4, II	6/15/75
39	Revised SIF Code - Vols. 1, 3, 4, 5	6/25/75
42	Change Readiness Levels - Vols. 2, III, IV	8/10/75

Configuration Index; Sample Format for Section I, Parts 1 and 2 (For a full description of format and content, see TM-5327/002.)

SECTION A. CPCI DEVELOPMENT RECORD

CPCI NO. AND NOMENCLATURE

CPCI No. 3021201
SSC Simulation Computer Program
for
Space Surveillance Center, System 437L

DEVELOPMENT SPECIFICATION NO.	ES04300.3021201
ISSUE DATE	21 Sep 69
AUTHENTICATION DATE	21 Oct 69

PRELIMINARY DESIGN REVIEW SCHEDULE	CRITICAL DESIGN REVIEW SCHEDULE
10 Jan 70 10 Jan 70 C	11 Apr 70 12 Apr 70 C
FUNCTIONAL CONFIGURATION AUDIT SCHEDULE	**PRODUCT SPECIFICATION SCHEDULE, SUBMITTAL**
~~17 Sep 70~~ 24 Sep 70 24 Sep 70 C	1 Aug 70 12 Aug 70 C
PHYSICAL CONFIGURATION AUDIT SCHEDULE	**CPCI QUALIFICATION SCHEDULE**
13 Sep 70 4 Oct 70 C	10 Sep 70 15 Sep 70 C
PRODUCT SPECIFICATION SCHEDULE, AUTHENTICATION	**CPCI QUALIFICATION CERTIFICATION DATE**
17 Oct 70 17 Oct 70 C	25 Oct 70

QUALIFICATION TEST DOCUMENTS

See Section III of this Configuration Index.

CONTRACTOR	CONTRACT NO.
SYSTEM DEVELOPMENT CORPORATION	F19628-70-C-0000

Configuration Index; Illustration of Format and Content for Section A, CPIC Development Record.

Figure 7. Configuration Index and Development Status.

SECTION I. CHANGE STATUS LISTING

ECP No.	CPCI No.	Title	Status	Comment
11-R1	SDC0324	Display Expansion Levels	A	SCN Approved
12-R2	SDC0324	Data Link Buffer Output	A	SCN Approved
15	SDC0327	Record Track Interrupts	A	
16	SDC0326	Suppress Tape SRN Message	X	Deferred for Study
18-R1	SDC0324	Process Simulated A-Link Data	A	SCN Distributed
18-1-R1	SDC0326	Process Simulated A-Link Data	A	SCN Distributed
18-2-R1	SDC0327	Process Simulated A-Link Data	A	SCN Distributed
20-R1	SDC0325	Modify Interrupt Priorities	X	Pending Interface Requirements
21	SDC0324	Data Monitor Control	P	
22	SDC0324	Positive Action Feedback	P	

STATUS INDICATORS:

P = ECP IS BEING PREPARED
S = ECP HAS BEEN SUBMITTED
A = ECP HAS BEEN APPROVED
D = ECP HAS BEEN DISAPPROVED
X = ECP HAS BEEN DEFERRED
I = ECP HAS BEEN IMPLEMENTED

SECTION II. CHANGE STATUS SUMMARY

[Summary information required for each ECP:]

a. ECP Number and Title
b. CPCI Number and Title
c. Brief Problem Summary
d. Description of Proposed Solution
e. Reference Documents
f. Preparation Status
g. Action Status
h. Implementation Status

Change Status Report; Summary of Total Content and Sample Format for Section I, Change Status Listing (For a complete description, see TM-5327/003.)

VERSION DESCRIPTION DOCUMENT

1. INVENTORY OF MATERIALS RELEASED
   - List of all items (tapes, cards, disks) covered by the VDD, by CPCI and version number. All related release documents for support items required to operate, load, or regenerate the released CPCI.
2. INVENTORY OF CPCI CONTENTS
   - List of all computer program instructions and data content released.
3. CLASS II CHANGES INSTALLED
   - Number, title, and issue date of each Class II CR; related SCN numbers and issue dates.
4. CLASS I CHANGES INSTALLED
   - Number, title, and issue date of each ECP; related SCN numbers and issue dates.
5. ADAPTATION DATA
   - When applicable: Identification of all unique-to-site (or mission) data contained in the item released.
6. INTERFACE COMPATIBILITY
   - Identification of other systems/CIs/CPCIs affected by incorporated changes, and present status.
7. BIBLIOGRAPHY OF REFERENCE DOCUMENTS
   - Listing of all pertinent documentation.
8. OPERATIONAL DESCRIPTION
   - Operational effects of Class I and II changes incorporated.
9. INSTALLATION INSTRUCTIONS
   - Methods to install and checkout the delivered version.
10. POSSIBLE PROBLEMS AND KNOWN ERRORS
    - Needs for further testing; status of problem resolutions.

Version Description Document (VDD); Summary of Contents (For complete instructions, see Section 10 of TM-5327/001.)

Figure 8. Change Status and Version Descriptions.

ENGINEERING CHANGE PROPOSAL (SHORT FORM) *(SEE MIL-STD-481 FOR INSTRUCTIONS)*

DATE PREPARED | ECP NO. | PROCURING ACTIVITY NO.

1. ORIGINATOR NAME AND ADDRESS | 2. MFR. CODE | 3. CLASS OF ECP | 4. JUST. CODE | 5. PRIORITY

6. SPECIFICATIONS AFFECTED		7. DRAWINGS AFFECTED		
MFR. CODE	SPECIFICATION/DOCUMENT NO.	MFR. CODE	NUMBER	REV.

8. TITLE OF CHANGE | 9. CONTRACT NO. & LINE ITEM

10. CONFIGURATION ITEM NOMENCLATURE | 11. IN PRODUCTION ☐ YES ☐ NO

12. NAME OF PART OR LOWEST ASSEMBLY AFFECTED | 13. PART NO. OR TYPE DESIGNATION

14. DESCRIPTION OF CHANGE

15. NEED FOR CHANGE

16. EFFECT ON ASSOCIATED EQUIPMENT

17. PRODUCTION EFFECTIVITY BY SERIAL NO. | 18. EFFECT ON PRODUCTION DELIVERY SCHEDULE

19. RECOMMENDED RETROFIT EFFECTIVITY | 20. ESTIMATED KIT DELIVERY SCHEDULE

21. ESTIMATED COSTS/SAVINGS

22. SUBMITTING ACTIVITY AUTHORIZING SIGNATURE | TITLE

23. APPROVAL/DISAPPROVAL

GOVERNMENT ACTIVITY | SIGNATURE | DATE

DD FORM 1 DEC 68 1693

GPO 1969 O—352-804

Figure 9. Format Illustrating Basic Information Required in the Engineering Change Proposal for a Computer Program Item.

COMPUTER PROGRAM
CLASS II CHANGE REPORT

ORIGINATOR System Development Corporation 2315 E. Pikes Peak Avenue • Colorado Springs, Colorado 80909				DATE
SPEC. NO., PART	VERSION NO.	CR NO.	REV.	CORR.
CPCI NOMENCLATURE				
TITLE OF CHANGE				
DESCRIPTION OF CHANGE				
JUSTIFICATION				
RELEASED BY	AUTHOR			
CLASSIFICATION APPROVAL				DATE

Figure 10. Sample Format for the Change Report (CR) Form.

DATE OF SUBMITTAL: _____ PROJECT IDENTIFIER: _____

SPR IDENTIFIER: _____

SPR TYPE: FR ☐ DC ☐ MC ☐ EF ☐ ED ☐ EC ☐

EMPLOYEE IDENTIFIER: _____

DATE OF PROBLEM DISCOVERY: _____ TIME OF DAY: _____

WORK IDENTIFIER IN PROGRESS: _____

STATUS: OPEN ☐ CLOSED ☐ PENDING ☐ OTHER ☐

PRODUCTS IMPACTED BY PROBLEM: _____

_____

_____

PRODUCTS USED WHEN PROBLEM OCCURRED:

Data Base Identifier: _____

Test Case Identifier: _____

Test Tool Identifier: _____

PROBLEM DESCRIPTION: _____

_____

_____

_____

_____

_____

DATE RECEIVED: _____ EMPLOYEE ASSIGNED: _____

Figure 11. Software Problem Report Form.

(Ref. Tor 269(4110-01) - 38 Reissue B)

DISCREPANCY REPORT FORM

MACHINE TYPE

TO: Computer Program Development Library c/o System Development Corporation
3000 Olympic Boulevard • Santa Monica, California 90406

CPDL CONTROL NUMBER _____

FROM:

Name _____ Phone _____

DATE LOGGED _____

ORIGINATOR'S NUMBER _____

Firm _____ Date _____

RECOMMENDED PRIORITY: ☐ HIGH PRIORITY ☐ MEDIUM PRIORITY ☐ LOW PRIORITY

Date or Model Required: _____ RESPONSIBLE ORGANIZATION _____

PROBLEM WITH: *(Check which is in error)*

☐ PROGRAM NAME _____ IDENT _____ MOD _____

TAPE/DISK NAME _____ MODEL(s) AFFECTED: _____

SUBSYSTEM INVOLVED _____

☐ DOCUMENT _____ M.S. _____

☐ EQUIPMENT _____

PROBLEM DESCRIPTION

ADDITIONAL DETAILS *(Describe or attach any materials necessary to specifically describe the problem)*

☐ ATTACHMENTS SUBMITTED

FINAL DISPOSITION: TO BE COMPLETED BY CPDL	FOR AIR FORCE USE
DRF Processed By _____ Date _____	DRF Processed By _____ Date _____

FOR ADDITIONAL COPIES CONTACT THE CPDL

FORM 42

Figure 12. Discrepancy or Error Report Form

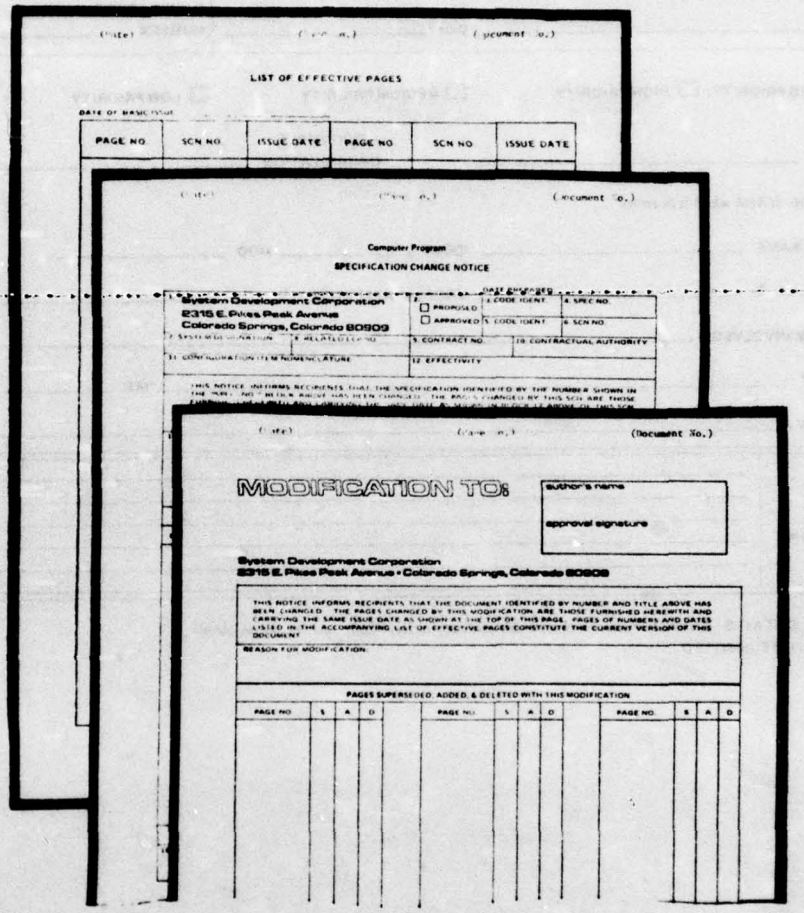


Figure 13. Special Front-Matter Pages for Document Maintenance (For a full description of these forms, see Section 7 of TM-5327/001.) [61].

and reinstates the error (DRF) in an active state.

Normally, there is not a great deal of information to be extracted from SCN or CN (see Figure 13). The Change Pages to the document contain the technical material and it seems redundant to repeat it all in the Specification Change Notice. Some short summary of what is transmitted is desirable, but the bulk of SCN's currently filed are cryptic. In general, this does not create great hardships for the configuration manager, but for the researcher there may be a blank between the filing and analysis of a modification request and the ultimate solution. In theory, ECP's are suppose to be revised to reflect all actions as the request is processed, but this does not always occur. A solution adopted by the 472M contract was to include copies of the final, revised ECP's and CR's with the SCN and Change Pages that transmit the implemented changes.

5.4 QUALITY CONTROL

Quality assurance for software projects is exercised through the specified reviews, audits and tests. Most projects do not have formal means of reporting review comment and resolve differences in conference. The only public record of specification errors or deficiencies is the difference between the original specification and the final approved document. There is seldom an official list of review criteria, and the CCB or project office depends upon the individual expertise of the reviewers to offset this deficiency. Some projects do file memoranda critiquing the specifications, and the CCB may issue Action Items that the project must respond to officially before the final approved changes are made. Directives may be issued to transmit CCB decisions.

Test documentation tends to be more formal than that for reviews and audits. The documents are normally controlled via the Configuration Index, maintained by SCN's, and baselined like other specifications. Hence, detailed testing criteria may be extracted. Although official records may not be kept on errors discovered during internal testing, when an independent test organization is employed, records are kept after delivery for integration and system test.

Both configuration control and quality control measures are often relaxed after delivery. However, a few systems do operate with a full range of configuration management procedures (the SAMSO SCF, for instance), and others have similar, more informal procedures.

5.5 SUMMARY

In general, when a project follows the full set of project and configuration management practices laid down by military standards and regulations, most of the information necessary to establish a proper repository is available. It is true that it would take a large amount of work to extract the desired information from reports and paper work, and put it into a machine-storable and retrievable mode. There is also a relatively low level of standardization in the data collected and in data collection forms and procedures. If one were going to try to create a central repository based on current data collection procedures, the attempt would probably fail due to a morass of unrelated data forms as have earlier attempts to build software development data bases. Many of the forms and procedures inherent in the current methods are required in order to coordinate the project and make management control work. To build a software data collection system around current manual procedures, many additional data requirements would have to be placed on contracts, new data collection forms generated, and new procedures developed both to extract the information from the existing procedures and to compile it into a coherent set of data.

6. DATA COLLECTION MONITOR SYSTEMS

If it is assumed that the analyses of data collection problems, data requirements and current military practices define the requirements for a data collection system, then existing systems may be examined to determine how well they meet these requirements. Among the monitor systems evaluated were SDC's IMPACT, MITRE Corporation's SIMON, BMDATC's Quantitative Data Base, IBM's Management Data Collection and Reporting System, and TRW's Software Reliability Study data collection procedures. Each of these systems have somewhat different objectives and scope of applicability, but each offer the management of software development some information on the management of the resources required to produce the software product and/or manage the quality of the product produced.

6.1 IMPACT [62]

In 1973 System Development Corporation initiated work on a project called the Software Factory. The SDC Software Factory is an integrated set of tools, data stores, and methodologies that provide a procedural approach to the successful development of software systems. It exists within a dynamic environment of varying customer and user requirements, large and small development projects, and a variety of programming disciplines. The objective of the Software Factory is to increase programmer productivity, reduce system development costs, and improve software reliability. As an adjunct to the Software Factory, work was initiated on a management tool called IMPACT that was designed to assist the project manager with performing his job. The techniques embodied in the tool were directly derived from the MIL-STD documents for configuration management on the project level. Regardless of the type of management procedures established on software development projects, however, IMPACT intends to provide project management with a means for management control and visibility.

An overview of the interaction of the Software FACtory Executive (FACE) with IMPACT is presented in Figure 14, the organization of the IMPACT data base is presented in Figure 15, and the IMPACT capabilities are presented in Figure 16.

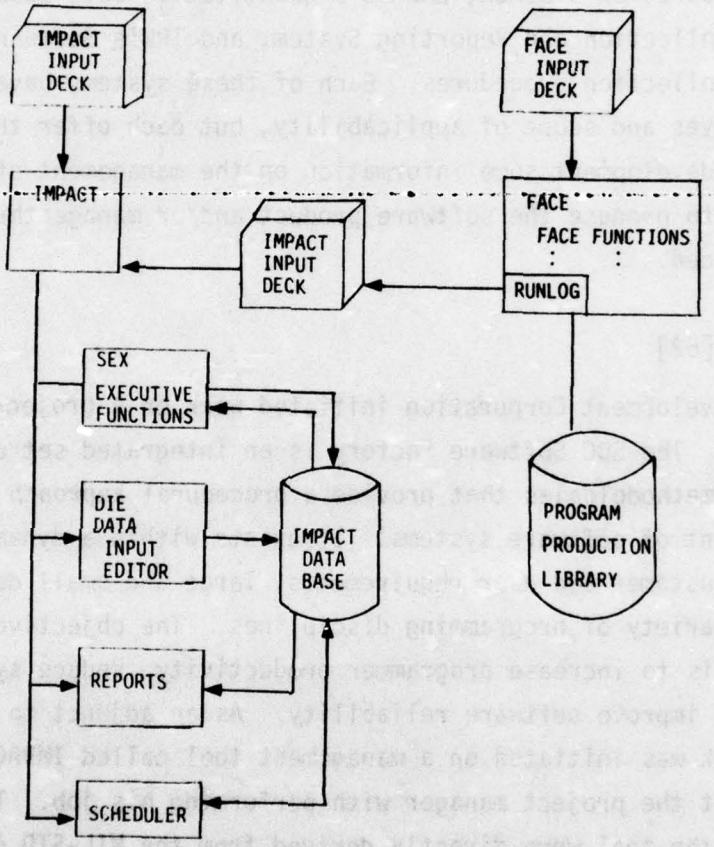


Figure 14. Software Factory/IMPACT

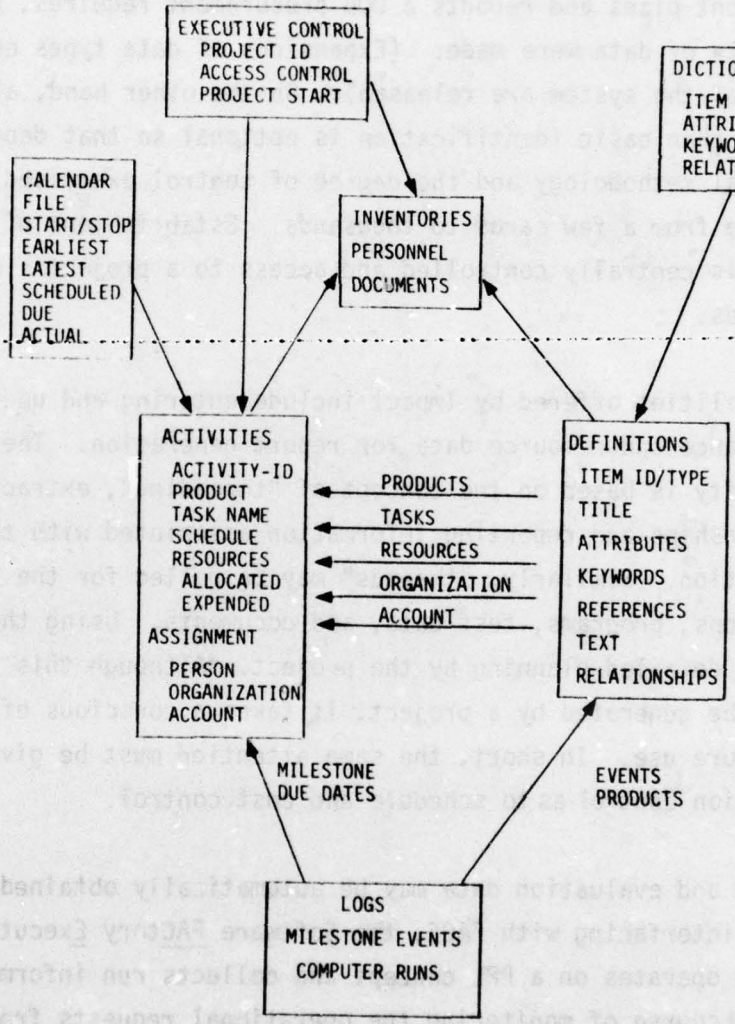


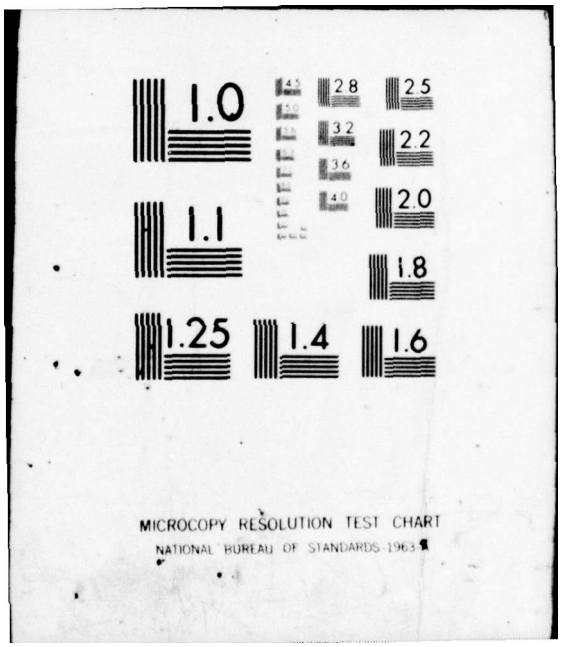
Figure 15. IMPACT Data Base.

AD-A036 116 SYSTEM DEVELOPMENT CORP SANTA MONICA CALIF F/G 9/2
SOFTWARE DATA COLLECTION STUDY. VOLUME II. AN ANALYSIS OF SOFTW--ETC(U)
DEC 76 N E WILLMORTH, M C FINFER F30602-75-C-0248
UNCLASSIFIED SDC-TM-5542/002/01 RADC-TR-76-329-VOL-2 NL

2 OF 2
AD
A036116

END

DATE
FILMED
3-77


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

The structure and capabilities of the IMPACT system are extensive with the result that its manual input requirements seem high. In seeking an integrated approach to project management information needs, as typified by the type of software development plans and reports a DOD procurement requires, provisions for a large variety of data were made. (Expansions of data types continue as planned versions of the system are released). On the other hand, almost all information other than basic identification is optional so that depending upon the managerial methodology and the degree of control exercised, manual inputs could range from a few cards to thousands. Establishment of a project in the data base is centrally controlled and access to a project's data is limited by keywords.

Some of the capabilities offered by Impact include entering and updating schedule, performance and resource data for report generation. The system's reporting capability is based on the concept of "threading", extracting a set of module relationships and reporting information associated with the extracted configuration. Similarly, "threads" may be pulled for the impact of changes to functions, programs, test data, and documents. Using these capabilities requires detailed planning by the project. Although this is information that must be generated by a project, it takes a conscious effort to record it for future use. In short, the same attention must be given to detail configuration control as to schedule and cost control.

Module statistics and evaluation data may be automatically obtained by the IMPACT system by interfacing with FACE, the Software FACtory Executive. This executive routine operates on a PPL concept and collects run information on each module in the course of monitoring the operational requests from users. The interface requirements between IMPACT and FACE are minimal; more or less data can be obtained as user needs increase for it. Note also that it is not necessary for IMPACT to run with FACE. These data could be input manually or by other automatic tools.

6.1.1 Environmental Characteristics

IMPACT does not specifically collect data of this nature. However, there is the ability to enter and update data relating to the individual project

attributes through specific reporting capabilities offered by the system. Identification of the following project characteristics can be made:

- Uniquely identify the project, its start date and control personnel
- Describe organization and structure
- Describe project tasks and activities
- Describe project resources and accounts
- Identify and inventory available personnel
- Describe project or system equipment and/or tools available
- Maintain records of changes

Addition project information may be available through references made to documents that further describe the project elements, such as organizational charters, job descriptions, and tool descriptions.

6.1.2 Project Performance Characteristics

Since IMPACT is designed to aid the project manager to better perform his job in visualizing project performance through reports, there are much data available to support research on productivity and progress. Some of the items relating to this include capabilities to:

- Define project activities and relate them to products and tasks
- Schedule project activities and record activity performance by actual starts and completions
- Constrain project schedules by milestone due dates and other event dates
- Record and compare resources allocated to and expended on an activity
- Record cost and schedule variance
- Investigate the impacts of limited resources on potential schedule performance
- Level peaks and valleys in resource allocations within available schedule slack time
- Assign responsibility for activities to persons, organizations and cost centers

- Record the documents authorizing an activity and reporting the results of it
- Provide retrieval of activity records by schedule dates and periods, products, tasks, personnel, organization, accounts, and resource assignments
- Reports activity anomalies in schedules, expenditures, and record completeness
- Provide computer operations log (with FACE, the Software Factory Executive). This information includes:
  1. Type of computer operation (compile, execute, etc.)
  2. Identifies module(s) and type of data set, e.g., source, involved in above operation
  3. Identifies time/date and amount of computer time used for operation
  4. Identifies personnel making computer run
  5. Gives a result indicator (good, warning, or bad)
- Provides module evaluation data with FACE, the Software Factory Executive. This information includes:
  1. Source card count
  2. Size of object module in bytes
  3. Number of instructions in module
- Retain an activities history to:
  1. Enable 'Trend' and 'Earned Value' charts
  2. Acquire statistics on frequency and impact of replanning
  3. Make periodic budget or expenditure tables
- Retain a log of personnel assignments for activities, including:
  1. Personnel efficiency ratings
  2. Personnel loading analysis
- Automatically acquire computer time expenditures and run results from the computer system
- Make 'Thread' oreinted-management reports - that is, specify a configuration member and relationship of a specified hierarchical depth, or number of flow steps, whose subelements would be retrieved and used to find all activities associated

with the thread members, and to report these with the configuration relationships depicted

- Provide schedule maintenance, including the reflection of actual performance (starts and completions) in the existing schedule

6.1.3 Product Configuration Characteristics

Much attention has been given to the design of configuration control procedures in the IMPACT System. Consequently, there is much data defining the system components and relationships. The capabilities IMPACT offers include the following:

- Provide for the identification of system configuration or representations at functional, design, and product phases in the project.
- Provide for describing the functional and structural relationships among system elements; especially the hierarchies of program, data, equipment, and function structures (decompositions).
- Provide for recording the baseline status of system elements.
- Provide for recording, describing and maintaining status accounts of all proposed changes to baselined products, including:
  1. Modifications (ECP)
  2. Problems (PR)
  3. Discrepancies (errors) (DRF)
- Relate the proposed changes to the configuration elements affected
- Record actual changes and associate these with the change proposals and elements changed, including:
  1. Specification Change Notice (SCN)
  2. Change Report (CR)
  3. Modification Transmittal (MTM)

- Provide for status accounting of changes, including due dates, approvals and implementations.
- Enable the detection of the impacts of proposed changes by following the recorded relationships between functions, program and data modules, tests, and documentation elements.
- Provide for the identification of system documentation, the decomposition of the document into its subsections, and cross-referencing of document elements to programs, data, functions, tests, modifications, and other configuration elements.
- Provide for the verification of actual program structure against planned. (At present, only the module-ID and module size are captured for verification; system cross-reference matrix not implemented.)
- Provide for configuration-oriented reporting i.e., arbitrary version compositions as well as baselined configurations, including:
  1. Configuration identification showing hierarchical structure.
  2. Configuration status showing documentation, modifications, corrections, and milestone status of each.
  3. Change status by class: ECP, PR, etc.
  4. Discrepancy status.

6.1.4 Quality Control

IMPACT offers the user the following quality assurance reporting capabilities:

- Identify and describe reviews and tests.
- For each product, define milestone dates for associated reviews and tests.
- Relate tests to the modules tested and the functions exercised.

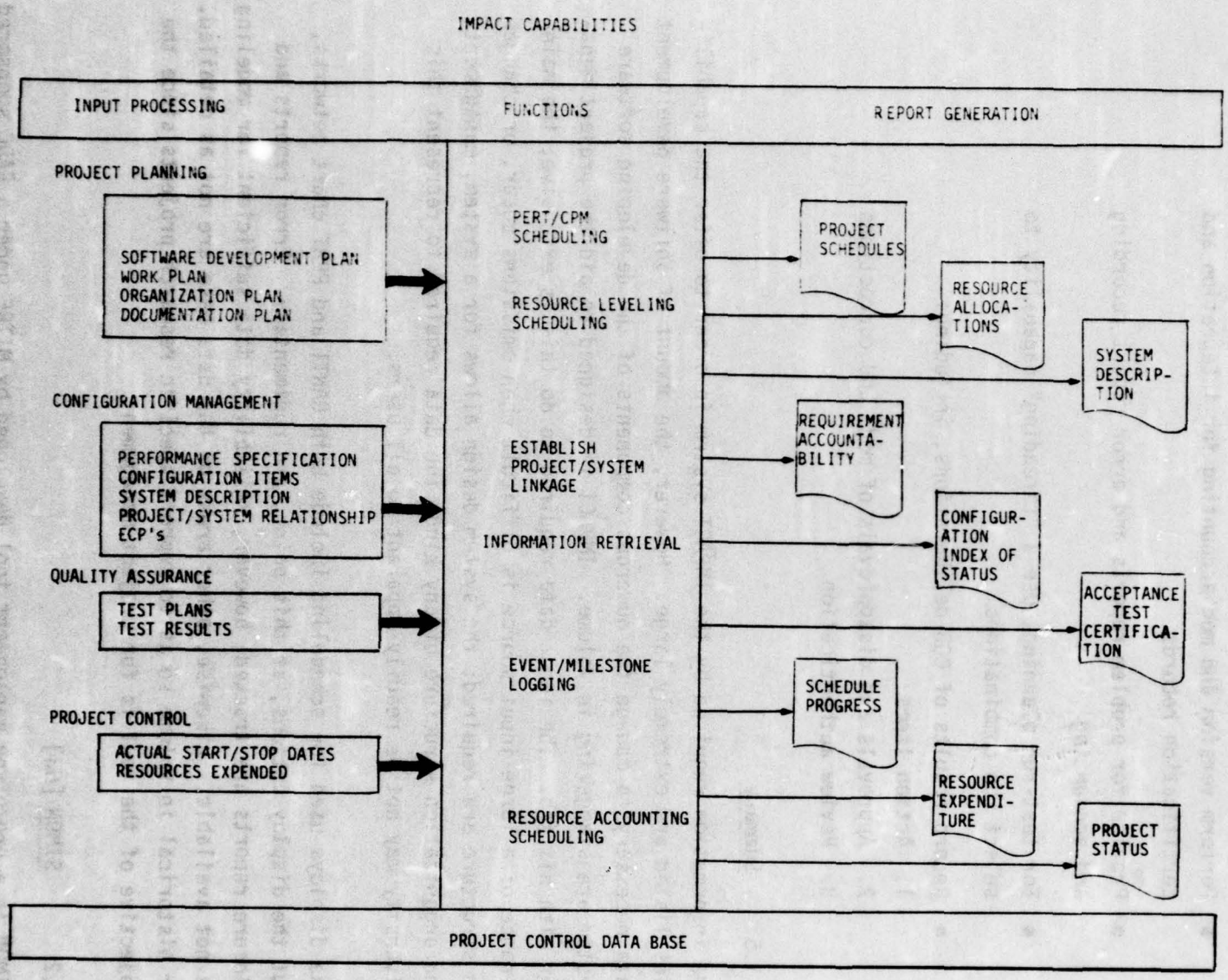


Figure 16. IMPACT Capabilities.

- Obtain immediate results of test runs from computer, and report current test status.
- Perform version and mod accounting for integration and certification records.
- Provide for problem reports and error report recording and accounting.
- For test-run planning, use a "threading" capability to select run combinations.
- Record results of CCB deliberations, including:
  1. Action Items
  2. Approvals and disapprovals of mods and corrections
  3. Review authentication

6.1.5 Summary

The information required by the IMPACT system in order to offer the capabilities listed are extremely large. However, the amount of software development data necessary to manage the numerous components of the developing software system are staggering in volume. IMPACT is designed to aid the project manager with his job. The input data required to do this is extensive; the maintenance of a large input source is difficult when omissions occur, or changes in structure are required; the system design allows for a system, management, and organization structure of any kind; the data required to represent this diversity may not be readily apparent to all users.

The displays used for scheduling include both GANTT and PERT chart networks, but the display options, at this point, are rudimentary. Error reports and closure reports are tracked; however, reliability data sufficient for modeling is not available. Likewise, other areas of the data base are not as detailed, or historical in volume so as to support specific research projects since the objective of the tool is for project management.

6.2 SIMON [63]

SIMON is a prototype management tool developed by MITRE under a RADC sponsored

contract. It is a tool to aid the management of developing software systems at the first-level programming group and its immediate manager. The principal purposes of the tool are 1) provide managerial visibility into the software development process and 2) collect data in an organized and timely way in order to support research on cost and reliability analyses. It appears that the major emphasis of this data collection monitor is to supply the project manager with reports on the system structure and data relationships as it evolves, with minimal emphasis on resource allocation and expenditure, and its management.

In order to perform its tasks, the SIMON System relies heavily on the correct and timely manual input for both the precompiler pass and the transaction pass. It is not clear that any of the input forms could be eliminated even if costs for manual preparation of input forms were a major factor in a specific programming project. In general, the collection forms appear to be fairly simple to fill out and submit.

The structure of the system components (see Figure 17), the data relationships, and the module evaluation data is obtained automatically by the precompiler, compiler and post compiler inputs to the transactor.

6.2.1 Environmental Characteristics

From examination of the contents of the records in the data base, it can be seen that SIMON collects the following environmental data:

- Identifies project by name, start and stop date
- Identifies total funding allocation, and partial funding for man-hours, computer dollars, file space, terminal hours, and other dollars
- Describes project personnel assignments
- Describes organizational structure and interfaces of software system

6.2.2 Project Performance Characteristics

As previously stated, the SIMON System is a tool to aid the project manager

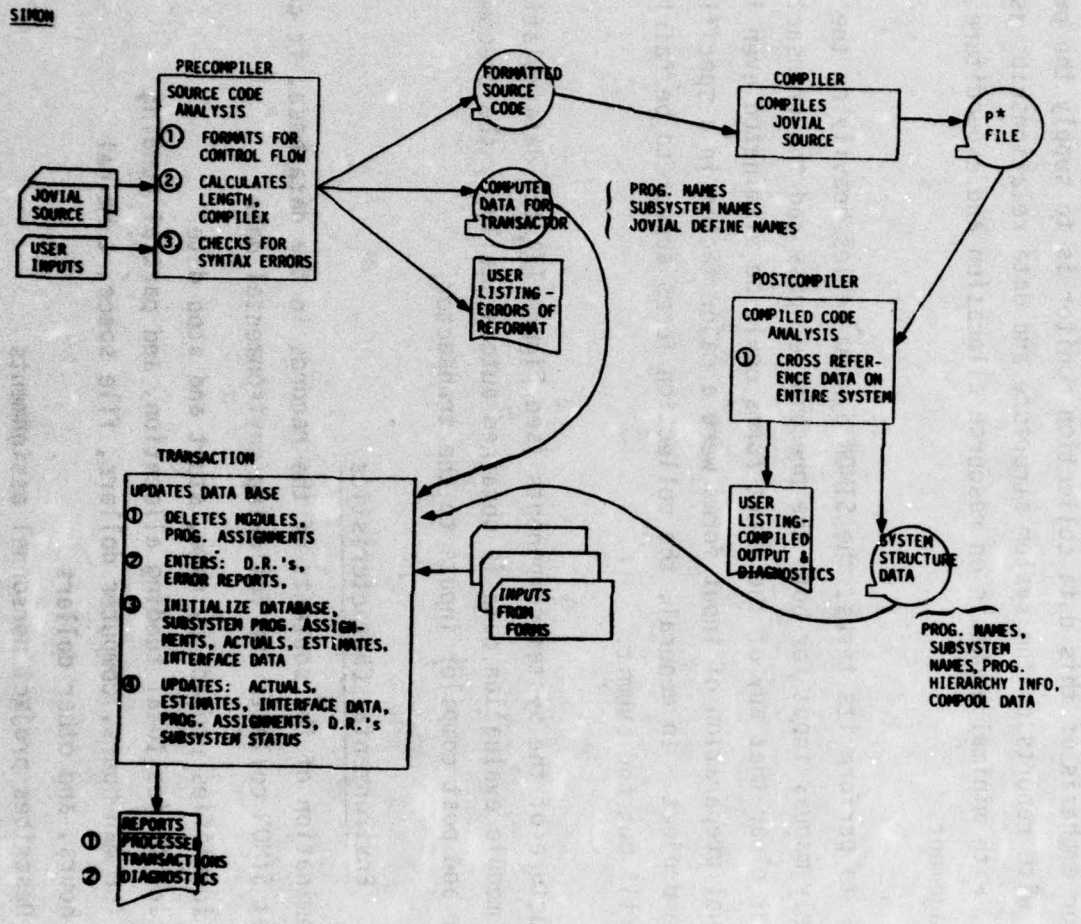


Figure 17. SIMON System Organization.

monitor the software through its development cycle. In so doing, it specifically collects enough data to:

- Schedule project assignments and record start and end dates.
- Record and compare resource allocated to and expended on system and subsystem design and test activities.
- Report resource expenditures (projected and actual) for person hours, terminal hours, main memory usage, computer dollars and file space.
- Report activity anomalies in schedules as manually input by both project manager and programmer.
- Reports projected resource overrun.
- Reports projected scheduling conflicts.
- Provides module evaluation data, including:
  1. Halstead length - count of operators and operands
  2. Number of program lines
  3. Number of statements
  4. Complexity rating, to be implemented
  5. Modules called
  6. Common variables referenced
  7. Files referenced, by name and type
  8. DEFINES referenced, by name
  9. Status of testing
  10. Number errors charged to module
- Provides computer operations status in reference to modules, including:
  1. Dates of first/last precompilations
  2. Dates of first/last compilations
  3. Total number of compilations
  4. Date of first "clean" compile
- Provides subsystem evaluation data, including:
  1. Status of design
  2. Testing status, including:
     1. Subsystems needed to complete testing

2. Driver needed and/or feasible for testing
3. Test plan status

3. Number and names of modules in subsystem and the testing status of each module.

6.2.3 Product Configuration Characteristics

The following reporting is available:

- Provides for the identification of system elements at the design and testing phases of the project.
- Provides for describing the structural relationships among system elements, including hierarchies of subsystems and modules, and data and file cross references.
- Provides for accounting of errors and discrepancies for subsystem and modules.
- Relates the errors to the system elements affected.
- Provides for the identification of system documentation on a subsystem level.
- Provides for the verification of actual module and subsystem size, and hierarchical structure against planned design.
- Provides a planning guide for the formation of system design changes.

6.2.4 Quality Control

SIMON provides the project manager with the following information:

- Identifies milestones of design and test for each system element.
- Identifies testing status and number of tests run for each module.
- Provides for error recording, including:
  1. Means error manifested itself
  2. Means of error diagnosis
  3. Complexity of error

4. Number of error occurrences
5. Time when error occurred

- Provides for discrepancy recording, including:
  1. Means of discrepancy detection
  2. Disposition of discrepancy
- Provides for error/discrepancy summations for the project for modules for each reporting period.
- Provide a mechanism for measuring system design disciplines, e.g., levels of abstraction when delineated by subsystem levels.

6.2.5 Summary

SIMON appears to be geared to a specific type of management procedure and for a specific type of software development project, limiting its applicability to differing projects.

For instance, the phases of software development are broken down into only 1) design; and 2) test. The implementation, or coding phase, is not specified as an independent activity to be included in scheduling or costing. Certainly an optimally finer breakdown of activities, as well as resource items, is an essential ingredient in any management tool. Not only does there appear to be an incomplete accounting of resources and expenditures, but the level of detail appears to be too coarse for accurate accounting and scheduling.

Configuration control procedures outlined by military organizations appear to be ignored. Configuration accounting is the process of monitoring the status of a module with respect to a set of specifications; all modifications to the module must also be reflected in the specifications and modifications to the specifications. Module history accounting consists of the original module source, the change cards used to produce each mod of a module, and whatever information is necessary to reconstruct a particular version of a module. The objective of module history accounting is to provide the facility a means by which one may revert to any prior state of a module.

Since one of the purposes of the SIMON System is to collect data for research into factors affecting software quality, the data being collected to support reliability analysis is insufficient in detail, such as the amount of effort to detect, isolate and install a correction. (SIMON requests time spent only "if significant".) Also, a finer delineation of type of error discrepancy for both reliability analyses and project management is desirable. (The Discrepancy Report Form and Discrepancy Update Form does not request such data as program name, subsystem involved, document referenced, recommended priority for repair, etc. It appears that these forms are a reporting mechanism only. The actual tracking of the error, its impact on the system, the cost of repair, the name and type of activity which originated it are not requested.) The phases of software testing have not been delineated with the result that error analysis for unit, integration and system testing cannot be determined. Since the JOCIT Compiler is a part of the SIMON System, a detailed history file could be collected, maintained, and reported. The project environment data collected is insufficient to identify characteristics of projects to support analysis of their impact on both costs and quality of software development.

6.3 BMDATC QUANTITATIVE DATA BASE [64]

System Development Corporation, in Huntsville, Ala., was contracted to establish procedures for the acquisition and maintenance of software development performance data received from BMDATC associate contractors. These contractors consisted of a set of Huntsville based companies supporting the Advanced Research Center. The procedures established were aimed at the acquisition of data obtained from the definition, design, production and test of software items constituting the BMDATC Software Development and Evaluation Technology (SDET) Program. The objectives of the data collection procedure and analysis were: 1) to provide measures of data processing technology effectiveness; and 2) to make software development cost estimates. Because of the type of research work being performed by the responding contractors and because the data collection procedure was an inter-company effort, the design of the data collection procedure was to establish a repository of data for analysis only. It was not intended to provide data for contract monitoring or project management.

The totally manual data collection procedure established consisted of two forms, the Software Development Data Form (SDDF) and the Software Modification Data Form (SMDF). The SDDF was submitted on a quarterly basis to SDC personnel. Development data prior to the establishment of the proceduralized collection date was estimated and subsequently submitted to SDC. (These estimated data reflected a time period ranging as far back as two years.) The SMDF was submitted upon completion of the modification by the person responsible for resolving the problem. It should be noted that this form was designed to report errors or modifications which exceeded one man-day's effort to resolve. One problem was to be reported per SMDF. All of the collection and compilation of the data obtained was manual. The data base was represented in two forms, the Accumulated Production Data and Accumuated Resources Used. These forms also included the estimated data reflecting work completed before establishing the procedure.

6.3.1 Environmental Characteristics

Although the BMDATC contractors are identified within the data base, little data supporting identification of unique project characteristics is made. The data collected:

- Identifies project and organization by name.
- Identifies reporting period.
- Identifies phase of software development life cycle that is currently being report on.
- Identifies staff experience of project personnel, including management.

6.3.2 Project Performance Characteristics

It should be noted that the data collected was not intended to be used for project management. The performance data forms, submitted on a quarterly basis:

- Records man-days expended on each software item activity, with relation to staff experience level.
- Records man-days expended on documentation produced.

- Records total amount of documentation produced in number of pages of text, math, and drawings.
- Records total number of documents written.
- Records total amount of source statements coded in MOL and POL.
- Records amount of logical, mathematical, control and data management, and I/O statements coded.
- Records amount of computer time used, with relation to staff experience level.

6.3.3 Product Configuration Characteristics

There was no data collected for the purposes of demonstrating the product configuration

6.3.4 Quality Control

The purpose of the BMDATC Data Base was not aimed at quality control. However, the data collected:

- Identifies software activity being performed at time of modification including:
  1. Requirement specification
  2. Software design
  3. Unit test
  4. Integration
  5. Validation test
  6. Acceptance & performance test
- Identifies type of modification, including:
  1. Requirement change
  2. Requirement error
  3. Design change
  4. Design error
  5. Coding error
  6. Documentation error, with specification of document title

- Records effort expended in man-days for:
  1. Detection and isolation of error
  2. Correction of error or installation of change
- Records amount of computer time used for:
  1. Detection and isolation of error
  2. Correction of error or installation of change
- Records number of source statements changed or added in MOL or POL with respect to the following types of instructions:
  1. Logical
  2. Mathematical
  3. Control and Data Management
  4. Input/Output

6.3.5 Summary

The project characteristics collected appear to be insufficient to allow analyses of the data processing technology effectiveness employed by each contractor. There is no form for obtaining this data, and although the ARC is familiar with each contractor, a finer level of detail of individual project characteristics is necessary. This form should contain such information as project personnel experience, programming methodology used, programming tools and aids used, etc.

The Software Development Data Form appears to lack specificity on productivity data in relationship to costs incurred. In order to obtain this type of data the reporting period must be more frequent than quarterly. Even a monthly reporting period may not be frequent enough to adequately measure productivity and cost data. Further, reliable, accurate information cannot be obtained by estimation or reconstruction. Also, the complexity or difficulty level of the specific task being reported should be estimated. (Obviously, an extremely difficult operating system written in a MOL impacts the productivity rate, and must be evaluated as such.) It also appears that the computer utilization costs are not sufficiently detailed so as to allow cost analysis.

The other collection vehicles, the Software Modification Data Form (SMDF), lacks detail supporting both analysis on technological effectiveness and costs incurred due to program errors and/or modification. Reevaluation of both collection forms and procedures is currently planned by the Advanced Research Center.

6.4 IBM MANAGEMENT DATA COLLECTION AND REPORTING [65]

IBM published the Management Data Collection and Reporting volume as one report of a series that addresses information and guidelines in the adoption and use of structured programming methodology under contract to RADC. The objectives of this specific report are: 1) to define data items and a methodology for collection of those data to aid the software project manager in performing his job; 2) to provide a plan for studying the data collection process by examining the data collected. A major impetus for the study performed by IBM was to understand better the software development process, the control and management of the process, and how structured programming technology impacts the development.

The data collection system includes estimated and actual data, as well as information obtained from a multi-level reporting scheme. Much of the data are used and cross referenced in several of the reports. It appears that the collected data (or collection vehicles) may be geared to an IBM-oriented software development process, and may not be applicable to a wide diversity of other software firms without some redefinition and expansion. The data are not intended to support project management, including configuration management and quality control, since they are after-the-fact- and summary data.

The proposed data collection system includes both manual and automatic means of acquiring data. The reporting level and method of collection are represented in Figure 18. The reports to be generated from the data require that the software development data be properly established and maintained.

6.4.1 Environmental Characteristics

The data supporting analysis of the project's environmental data:

Data Classes and Types / Collecting and Updating Functions	Automatically Collected	Project Personnel Supplied	Collection Level(s)	Automatic Collecting During Development – Phases – Definition	Design	Implementation	Evaluation	User Updating During Development – Phases – Definition	Design	Implementation	Evaluation	Comments
PLAN DATA												
Project Environment		X	System and Subsystem					X	X	X	X	Most update activity will take place during the Design and Implementation Phase
Module Description		X	Program						X	X		
Resource Cost		X	Subsystem					X	X	X	X	
Computer Utilization		X	System						X	X		
Program Production		X	Program						X	X		
ACTUAL DATA												
Project Environment		X	System and Subsystem					X	X	X	X	
Module Description	X	X	Unit and Program			X	X			X	X	Approximately 80% of the data items can be automatically collected
Resource Cost		X	Subsystem					X	X	X	X	
Computer Utilization		X	Job						X	X		
Program Production	X	X	Unit and Program			X	X			X	X	Approximately 50% of the data items can be automatically collected

Figure 18. Data Class/Type Summary [65].

- Describes system structure and complexity.
- Identifies project and subsystem resources.
- Identifies personnel characteristics and experience.
- Identifies working environment.
- Identifies customer experience.
- Identifies travel requirements.
- Identifies structure and complexity of data base.

6.4.2 Project Performance Characteristics

The following reports on project performance information:

- Provides for determining the system status of elements during implementation and evaluation phases.
- Provides for determining the update activity and reasons. for making updates to system components during implementation and evaluation phases.
- Provides for determining the update activity and current status of a program during implementation and evaluation phases.
- Provides for the monitoring, optimizing and allocating of computer testing time during the design and implementation phase for system elements.
- Provides for monitoring actual amount of system documentation produced against estimated.
- Provides for monitoring the system construction during its development.
- Provides cost information, both estimated and actual, on resources expended current to the reporting cycle.
- Provides subsystem and project progress information based on estimated vs. actual data.
- Provides system technical information, both estimated and actual, current to the reporting cycle.

6.4.3 Product Configuration Characteristics

The data collection system provides data sufficient to verify actual system

implementation against system design.

6.4.4 Quality Control

The information to aid the project manager with quality:

- Provides information for monitoring the modification status for control, modules, programs, subsystems, system.
- Identifies number of system tests planned and executed to date.
- Provides an error reporting hierarchy for programs, subsystem and system including Specification Implementation Errors and Specification Development Errors.
- Provides efficiency improvements and documentation made for programs, subsystems and system.

6.4.5 Summary

Since data collection vehicles were not associated with the data collection system proposed, a complete evaluation of the data collection system cannot be made. It appears that error reporting and analysis is insufficient if management is to have an impact on the reliability of the system produced, as well as determining the effectivity of structured programming technology on the reliability of software. There is no facility to monitor problem reports or closure reports. Such items as error impact on system components, origination of error, costs incurred on error detection, isolation and correction must be included. Module history accounting is not provided for, although there is provision for automatically updating the module number of each program unit. Configuration control procdures outlined by military organizations are not monitored. The identification of the system configuration items at various phases of software development are not specified, and there appears to be no provision for identifying milestones in the software development process.

The programming support library (PSL) concept, a valid and essential element in contributing to the success of the data collection system must be evaluated. In order to determine its effectivity, specific data must be collected on the user procedures, human factors considerations, and costs of maintaining

the PSL. (The size of the software project may not be able to support the personnel necessary for the PSL. and alternatives should be examined as well.)

Schedule maintenance and schedule variances are not delineated, and consequently, it would be difficult to evaluate the impact of variances in schedule on resources. Also, personnel assignments, activities, and schedules are not specified or monitored. Project activities are not related to deliverable end-items, which may influence analysis on productivity of project personnel, as well as overall project performance.

6.5 TRW SOFTWARE RELIABILITY STUDY [66]

The research work TRW performed on software reliability and quality under contract to RADC contains a detailed study of software error types, techniques for finding those errors, and recommendations for improving reliability. The objectives of the study include:

1. Examine available software structure and characteristics which, when analyzed, will contribute factors for reliability description and/or prediction.
2. Define improved error collection methods.
3. Define error categories.
4. Recommend techniques for improving software reliability, and early error detection.
5. Evaluate reliability models.

TRW used data from four large software development projects to support this study, as well as examining many methods of reliability modeling.

6.5.1 Conclusions of the Study

The reliability study provides an insight into the benefits derived from collecting a detailed sample of error data. The analyses performed have the potential for aiding both present and future software development projects. These collection and analyses processes, however, are not obtained without a substantial amount of cost and effort, as well as the support of all project personnel.

The collecting of error data may be improved by the following recommendations:

1. Establishing adequate procedures for the problem report and problem closure.
2. Establishing procedures for standards and formats for data collection, with proper regard to configuration management and reliability data requirements.
3. Applying general purpose reliability data collection tools.
4. Allocating "dedicated" manpower to the collection process.
5. Initiating data collection procedures early in the development cycle.

The categorization of errors can be broken into sub-categories, one for classifying the symptom and one for classifying the specific cause. These categories can then be further delineated, according to the following observations:

1. The "fixer" of the error should be responsible for the error category assignment.
2. The error categorization should be:
   a. Done concurrently with closure of the error.
   b. Based on both the problem and closure reports.
3. Lengthy error categories are difficult to use and require practice to be effective.
4. Education of data collection personnel as to the objectives and precedures of the effort is necessary.

Software project and module characteristics necessary for reliability analysis are of two types:

1. Measurable structural characteristics, including size, interface descriptions, data base use, language elements use, data handling, etc.
2. Subjective evaluation characteristics, including difficulty, type or routine, complexity, etc.

Recommended techniques for improving software reliability include:

1. Examination of error data, including historical error rates, the criticality of errors found, and the concentration of errors in source code, can indicate areas of poor design. In such cases, action can be taken to eliminate the problems before delivery of the product to the customer.
2. New software technology, including structured programming, and test tools to dynamically record execution frequency, impact error rates.

6.5.2 Summary

The TRW study was primarily directed at examining components of software reliability. However, as the interim report notes, there were several important data points that were not collected or analyzed during the course of the study – more extensive reliability modeling data, such as total CPU operating time of the software, total elapsed debugging time, number of tests run, number of errors found per test, etc. While this information is specific to reliability modeling, it should be provided for in software operations reports on the contigency that modeling is to be done. Design error information, such as when the error was discovered and the consequence of the error, including amount of code change, complexity of change, number of modules impacted, etc., is also important for reliability analysis.

There is little data to support analysis of the number and severity of errors generated as a result of correcting a previous error. Also, information as to amount of effort needed to correct an error, e.g., number of computer runs, total CPU time, total resource expenditures, and man-hours spent, is needed for program error analysis. More extensive testing information, e.g., number of test cases run, number of errors per test case, amount of code tested per test case, and type of testing methodology used is necessary for the evolution of better test tools.

6.6 CONCLUSION OF COMPARATIVE EVALUATION

Despite the difficulty of comparing data collection systems designed to meet widely differing objectives, several conclusions can be reached. There is no one system that provides a data collection procedure and data base structure to meet both research needs and project performance monitoring. The collection systems reviewed are geared to specific project dependent characteristics and do not provide flexibility in collecting data from diversified development projects. While there are problems inherent in attempting to collect and maintain data for a software project of any size and structure, as demonstrated by the management tool IMPACT which requires valid, well-defined plan data, it appears that such a requirement is necessary as it pertains to the RADC repository specifications. In order to analyze effectivity of programming methodologies and tools, system structure and data relationships are needed. A tool, such as SIMON, allows the manager to evaluate the construction of the evolving system automatically, but this large amount of data requires thorough and timely examination by a technical person in order to be used effectively. These data may be so detailed and depend so heavily on user inputs as to be virtually useless in a repository without summarization. Both planned and actual values need to present for some parameters; a method that collects "old" data necessarily collects subjective estimates and invalidates the project's data base. The Quantitative Data Base experience has demonstrated the need to collect data in a proceduralized format on a timely basis. A project library, such as IBM's PSL, appears to offer a systematic method of maintaining software project data in an organized and controlled method, requiring a significant commitment of personnel and computer resources to the data collection process. Reliability data sufficient in quantity and detail to support reliability modeling and indicate the technological direction in which to proceed in the development of new tools can be time consuming and costly for project personnel as demonstrated by the TRW study.

(The reverse of this page is blank)

BIBLIOGRAPHY

SDC Studies

[1] TM-903/000/02, Management of Computer Programming for Command and Control Systems. K. Heinz, N. Claussen, V. LaBolle, System Development Corporation, 8 May 1963.

TM-1021/002/00, A Description of the Computer Program Implementation Process, L. Farr, 25 February 1963.

TM-1021/003/00, A Description of the Computer Program Implementation Process: A Process Flow, R. E. Bleier, 9 May 1963.

TM-1447/000/02, Factors that Affect the Cost of Computer Programming, L. Farr, B. Nanus, System Development Corporation, 30 June 1964.

TM-1447/001/00, Research into the Management of Computer Programming: A Quantitative Analysis, L. Farr, H. Zagorski, System Development Corporation, 31 August 1964.

TM-1447/002/00, A Summary of an Analysis of Computer Programming Cost Factors, L. Farr, H. Zagorski, 25 January 1965.

TM-1603/000/00, Frequency Analysis of Machine Instructions in Computer Program Systems, R. E. Bleier, 19 November 1963.

SP-1747, Estimation of Computer Programming Costs, V. LaBolle, System Development Corporation, 14 September 1964.

SP-1934, Critical Management Points, V. LaBolle, System Development Corporation, 12 February 1965.

SP-2059, Research in the Management of Computer Programming, G. Weinwurm, System Development Corporation, 1 May 1965.

TM-2314, Planning Guide for Computer Program Development, L. Farr, V. LaBolle, and N. E. Willmorth, System Development Corporation, 10 May 1965 (Revised 1969).

TM-2704/000/00, Research into the Management of Computer Programs: Some Characteristics of Programming Cost Data for Government and Industry, A. E. Nelson, System Development Corporation, 15 November 1965.

TM-2712/000/00, Research into the Management of Computer Programming: A Transition Analysis of Cost Estimating Techniques, G. Weinwurm, H. Zagorski, System Development Corporation, 12 November 1965.

TM-2918/000/00, Development of Equations for Estimating the Costs of Computer Program Production, V. LaBolle, System Development Corporation, 5 April 1966.

TM-3026/000/01, Current Results from the Analaysis of Cost Data for Computer Programming, T. Fleishman, System Development Corporation, 26 July 1966.

TM-3225/000/00, Management Handbook for the Estimation of Computer Programming Costs, E. A. Nelson, System Development Corporation, 31 October 1966.

TM-2222/Volumes 000-021/00, System Programing Management, N. E. Willmorth, System Development Corporation, 1965.

PRC Studies

[2] ESD-TR-66-673, Air Force ADP Experience Handbook (Pilot Version), A.J. Gradwohl, et al, PRC, December 1966.

ESD-TR-66-672, Primer for Air Force ADP Experience Handbook (Pilot Version), A. J. Gradwohl, et al, PRC, December 1966.

ESD-TR-66-671, Phase II Final Report on the Use of Air Force ADP Experience to Assist Air Force ADP Management, A.J. Gradwohl, G. S. Bechwith, S. H. Wong, W. O. Wootan Jr, PRC, 1966
Volume 1 - Summary, Conclusions and Recommendations
Volume 2 - Phase II Activities
Volume 3 - Phase III Concepts and Plan

[3] Reifer, D. J. Computer Program Verification/Validation/ Certification, TOR-0074(4116)-5, Aerospace Corporation, May 1974

[4] Weinberg, G. M., The Psychology of Improving Programming Performance, Datamation, November 1972.

[5] Brooks, F.P., Jr., The Mythical Man Month, Addition-Wesley, 1975.

[6] Grant, E.E. and Sackman, H., An Exploratory Investigation of Programmer Performance Under On-Line and Off-Line Conditions, IEEE Transactions on Human Factors in Electronics, March 1967.

[7] Reinstedt, R.N., Results of a Programmer Performance Prediction Study, IEEE Transactions on Engineering Management, December 1967.

[8] Carlson, W.M., A Management Information System Designed by Managers, Datamation, May 1967.

[9] Gibson, C.F. and Nolan, R.L., Managing the Four Stages of EDP Growth,

[10] Gholson, R.K., Managing Programming Productivity, Data Exchange, March 1974.

[11] Ryle, B.L., An Engineering Approach to Software Configuration Management, IEEE Transactions on Aerospace and Electronics Systems, November 1967.

[12] Schlight, R.J., A Functional Approach to Software Management, Proceedings of the Aeronautical Systems Software Workshop, AFSC,

[13] Weinwurm, G.F., On the Economic Analysis of Computer Programming, in On the Management of Computer Programming, Aeurbach, 1970.

[14] Mealy, G.H., Farker, S.J., Morenoff, E., Sottley, K., Program Transferability Study, RADC, 1968.

[15] Estell, R.G., Software Life-Cycle Management, in 3rd Annual DR&CG Seminar: Software Life-Cycle Management Techniques, U.S. Army Proving Grounds, October 1974.

[16] Gerloff, E.A., Performance Control in Governgment R&D Projects: The Measurable Effects of Performing Required Management and Engineering Techniques, IEEE Transactions on Engineering Management, February 1973.

[17] Baker, F.T., System Quality Through Structured Programming, Proceedings 1972 FJCC,

[18] MIL-STD 480, Configuration Control - Engineering Changes, Deviations and Waivers.

[19] MIL-STD 483, Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs.

[20] MIL-STD 490, Specification Practices.

[21] MIL-STD 499, System Engineering Practices.

[22] MIL-STD 881, Work Breakdown Structures for Defense Material Items.

[23] MIL-STD 1521, Technical Reviews and Audits for Systems, Equipment, and Computer Programs.

[24] Proc. Int. Conference on Reliable Software, SIGPLAN NOTICES, June 1975.

[25] Management Data Collection and Reporting, RADC-TR-74-300, Volume IX, IBM, October 1974, (A008640).

[26] Weinwurm, G.F., On the Economic Analysis of Computer Programming and the Challenge to the Management Community, in On the Management of Computer Programming, Auerbach, 1970.

[27] Schlight, R.J., A Functional Approach to Software Management, in Procs. of the Aeronautical Systems Software Workshop, AFSC, July 1974.

[28] Wolverton, R.W., Paradoxes in Management: Software Standards and Procedures, in Proc. Aeronautical Systems Software Workshop, AFSC, July 1974.

[29] Some Thoughts on Standard Measures of Performance, EDP Performance Review, January 1975.

[30] Hansen, P.B., Operating System Principles, Prentice-Hall, 1973.

[31] Kernighan, B.W. and Plauger, P.J., The Elements of Programming Style, McGraw-Hill, 1974.

[32] Van Tassel, D., Program Style, Design, Efficiency, Debugging, and Testing, Prentice Hall, 1974.

[33] Mathis, N.S. and Willmorth, N.E., Software Milestone Measurement Study, TD-285, NELC, November 1973.

[34] Schwartz, J.I., Construction of Software: Problems and Practicabilities in Practical Strategies for Developing Large Software Systems, Addison-Wesley, 1975.

[35] Wolverton, R.W., The Cost of Developing Large-Scale Software, IEEE Trans Computers, June 1974.

[36] Hartwick, R.D., Acquisition Management Based upon Case Histories in Proc. Aeronautical Systems Software Workshops, AFSC, July 1974.

[37] Keider, S.P., Why Projects Fail, Datamation, December 1974.

[38] Joseph, A.M., Time Estimates for Computer Programming, Retail Central, December 1973.

[39] King, W.R. and Wilson, T.A., Subjective Time Estimates in Critical Path Scheduling - A Preliminary Analysis, Management Science, January 1967.

[40] Thambain, H.J. and Wilemon, D.C., Diagnosing Conflict Determinants in Project Management, IEEE Trans. Eng. Mgt., February 1975.

[41] Corrigan, Ann. Results of an Experiment in the Application of Software Quality Principles, MTR-2874.

[42] McGregor, D., Do Management Control Systems Achieve Their Purpose, Mgt Rev, February 1967.

[43] Fleck, R.A. and Hodge, B., People: Hidden Asset or Liability, Data Management, April 1975.

[44] Ryle, B.L., An Engineering Approach to Software Configuration Management, IEEE Trans. Aerospace and Electronics Systems, November 1967.

[45] Walker, G.A., Life Cycle Cost/System Effectiveness: Evaluation and Criteria D180-176484, Boeing Company, December 1973.

[46] Summary Notes of a Government/Industry Software Sizing and Costing Workshop, ESD, Hancomb AFB, October 1974.

[47] Ellingson, O. E., Historical Analysis of SCF Computer Program Changes as a Guide to Establishing Quality Control Measures, TM-2287, System Development Corporation, March 1966.

[48] Tucker, A.E., The Correlation of Computer Program Quality with Testing Effort, TM-2219, System Development Corporation, January 1976.

[49] Aron, J.D., Estimating Resources for Large Programming System IBM, FSC-69-5013.

[50] Bakkegard, I.E., Quantitative Data Base, TM-HU-194/000/00, System Development Corporation, April 1975.

[51] Sharp, W.F., The Economics of Computers, Col. University Press, New York, 1972.

[52] Johnson, R.A., Newell, W.T., and Vergin, R.C., Operations Management, Houghton-Mifflin, New York, 1972.

[53] Airman's Pay Guide, U.S. Air Force Recruiting Service Directorate of Advertising, Oct. 1, 1975.

[54] Military vs Civilian Pay: Official Report, U.S. News and World Report, Dec. 8, 1975.

[55] Salary Supplement to Examination Announcements, U.S. Civil Service Commission, Feb. 1, 1976.

[56] The Bureau of National Affairs, Policy and Pay Practices, 1974.

[57] Schosky, D.P., D.P. Salary Survey, Datamation. Jan. 1976.

[58] AFSCP/AFLCP 173-5, AMCP 37-5, NAVMAT P5240, "Cost/Schedule Control Systems Criteria Joint Implementation Guide". March 31, 1972.

[59] AFR-800-14, The Management of Computer Resources, Vol. I, 10 May 1974; Vol. II. Aug. 1975.

[60] Report No. TDR-269(4110-01)-38, Reissue C, Change Control Procedures for Automatic Data Processing Systems (ADPS), USAF Space and Missiles System Organization, AFSC, 1 February 1972.

[61] Searle, L.V., Computer Program Configuration Management, TM-5327, 5 June 1974., Vol. 000, Introductory Guide to Computer Program Configuration Management; Vol. 001, Policies and Internal Procedures; Vol. 002, Guide to the Configuration Index; Vol. 003, Guide to the Change Status Report.

[62] Willmorth, N.E., "Integrated Management, Project Accounting and Control Techniques", System Development Corp., August 1975.

[63] Corrigan, A.E., et al, "Specifications for SIMON, a Software Implementation Monitor:, MTR-3056, MITRE Corp., Bedford, Massachusetts, July 1975.

[64] Bakkegard, I., "Quantitative Data Base" TM-HU-195/000/00, System Development Corp., Santa Monica, Calif., April 1975.

[65] Smith, Ronald L., "Management Data Collection and Reporting", Vol. IX in Structured Programming Series, IBM, Gaithersburg, Maryland, Oct., 1974.

[66] TRW Systems Group, "Software Reliability Study", Interim Report, RADC-TR-74-250, October 1974, (787784).

[67] ESD-TR-75-85, An Air Force Guide for Monitoring and Reporting Software Development Status, Sept. 1975, ESD-TR-75-91. Software Acquisition and Standards, October 1975.

[68] ESD-TR-75-365, An Air Force Guide to Contracting for Software Acquisition, January 1976.

☆U.S. GOVERNMENT PRINTING OFFICE: 1977-714-025/84

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.